Contextualisation, Visualisation et Evaluation en Apprentissage Non Supervisé

Laurent Candillier

THÈSE

pour l'obtention du titre de

DOCTEUR en INFORMATIQUE

de l'Université Charles de Gaulle - Lille 3

soutenue par

Laurent Candillier

avec le soutien de la société Pertinence, Paris

et de l'Association Nationale de la Recherche Technique

Contextualisation, visualisation et évaluation en apprentissage non supervisé

Date de soutenance : le 15 septembre 2006

Jury :

Prof. Edwin Diday	Université Paris Dauphine	Président, Rapporteur
Prof. Gérard Govaert	Université de Compiègne	Rapporteur
Prof. Rémi Gilleron	Université de Lille 3	Directeur de thèse
M. Conf. Isabelle Tellier	Université de Lille 3	Examinateur
M. Conf. Fabien Torre	Université de Lille 3	Examinateur
Chercheur Olivier Bousquet	Société Pertinence	Examinateur

Aucune règle n'existe,
les exemples ne viennent
qu'au secours des règles
en peine d'exister.

André Breton

Remerciements

Je tiens tout d'abord à remercier tout particulièrement Fabien Torre, sans qui cette thèse n'aurait jamais pu se dérouler dans de bonnes conditions, qui a toujours été très présent et a enrichi ces travaux de manière conséquente.

Merci à Isabelle Tellier qui a également été très présente tout au long de cette thèse et a permis l'enrichissement de ces travaux, et à Rémi Gilleron qui a accepté d'être directeur de ma thèse et m'a accueilli dans son équipe. Merci aussi à Dominique Gonzalez, Aurélien Lemay, Patrick Marty, Joachim Niehren, Philippe Preux et Marc Tommasi qui m'ont également aidé à avancer, et plus généralement à l'ensemble de l'équipe GRAppA de l'université de Lille 3, qui réussit à allier bonne entente et travail rigoureux, et dans laquelle j'ai passé des années de recherches très enrichissantes.

Je tiens également à citer l'Association Nationale de la Recherche Technique (ANRT) et la société Pertinence qui ont participé au financement de cette thèse par le biais des bourses CIFRE. Ces moments passés au sein de l'entreprise ont eux aussi été très enrichissants pour moi. Je remercie en particulier les membres de l'équipe de Recherche de la société, Olivier Bousquet, Alex Fleischer, Jean-Marc Kerisit, Seydina Ndiaye, Benjamin Rey et Skander Soltani.

Je remercie aussi Edwin Diday et Gérard Govaert qui ont bien voulu accepter la charge de rapporteurs de ces travaux. Les remarques pertinentes qui m'ont été faites ainsi que l'opportunité qui m'a été donnée d'améliorer encore cette thèse après la soutenance ont été, je pense et j'espère, très bénéfiques. Merci aussi à Edwin Diday d'avoir présider ce jury.

Enfin je tiens également à remercier Michèle Sebag et Catherine Trottier qui ont aussi contribué à l'avancée de ces recherches, ainsi que les différents rapporteurs anonymes de mes travaux pour leurs remarques et références pertinentes qui m'ont permis d'avancer.

Merci à ma famille et mes amis pour leur soutien et leur présence.

Merci au lecteur intéressé par mes travaux.

Laurent

Table des matières

Chapitre 1

Introduction

1.1 L'extraction de connaissances dans les bases de données

Depuis la venue de l'informatique, l'ensemble des données stockées sous forme numérique ne cesse de croître de plus en plus rapidement partout dans le monde. Les individus mettent de plus en plus les informations qu'ils possèdent à disposition de tous via le web. De nombreuses entreprises, en particulier la plupart des grands magasins, récoltent de plus en plus d'informations sur leurs clients et leurs comportements. De nombreux processus industriels sont également de plus en plus contrôlés par l'informatique. Les résultats d'analyses médicales sont aussi de plus en plus régulièrement conservés pour être analysés. Et de nombreuses mesures effectuées un peu partout dans le monde, comme par exemple les mesures météorologiques, remplissent aussi d'importantes bases de données numériques.

Ainsi, de nos jours, non seulement le nombre de données stockées numériquement est très important, mais le type de ces informations est également très varié. Le web est un exemple très présent aujourd'hui d'espace regroupant des données très nombreuses, diverses et variées : textes structurés ou non, images, sons, films, etc. Les bases de données clients, les données extraites de processus de production, les résultats d'analyses médicales ou les bases de données de mesures mondiales peuvent elles aussi contenir un nombre important d'informations hétérogènes : données numériques, catégorielles, courbes, etc. Il existe dès lors un très grand intérêt à développer des techniques permettant d'utiliser au mieux tous ces stocks d'informations, afin d'en extraire un maximum de connaissance utile.

Sur le web, il s'agit par exemple de comprendre au mieux le contenu des pages web et des requêtes utilisateurs afin de fournir à ces derniers l'information ciblée la plus pertinente possible et de la manière la plus compréhensible possible. Dans le cas de bases de données clients, il peut s'agir de comprendre au mieux les comportements des clients afin de leur faciliter l'accès aux produits qui les intéressent. Quant aux données issues de processus de production, il existe un grand intérêt à en extraire un maximum de connaissances afin d'en déduire des bonnes pratiques pour optimiser la production. En ce qui concerne les résultats d'analyses médicales, leur étude peut par exemple aider

à mieux détecter les patients à risque pour certaines maladies, permettant ainsi de prévenir plutôt que de guérir. Dans le cas des données météorologiques, leur analyse peut aider à mieux comprendre les phénomènes généraux qui régissent le climat, afin, par exemple, d'anticiper les phénomènes extrêmes et d'agir en conséquence pour les populations concernées.

L'objectif général de l'*apprentissage automatique* est d'extraire automatiquement de telles connaissances en se basant sur des exemples, c'est-à-dire sur un ensemble limité de données disponibles. Ce problème se décline en plusieurs variantes, en fonction des informations disponibles sur le problème traité :

- l'apprentissage supervisé,
- l'apprentissage non supervisé,
- l'apprentissage semi-supervisé,
- et l'apprentissage partiellement supervisé.

Mais avant de présenter ces différentes approches, nous introduisons d'abord différentes manières de représenter les données.

1.2 Représentation des données

Généralement, on représente les données sous forme tabulaire. Les lignes du tableau correspondent alors aux *exemples*, les colonnes aux *attributs* qui les caractérisent, et les cases du tableau aux valeurs des exemples sur ces attributs. On parle ainsi classiquement de représentation des données par *attribut-valeur*, ou sous forme matricielle.

Le tableau 1.1 représente par exemple un ensemble de voitures définies par un ensemble de caractéristiques : leur identifiant, le carburant qu'elles utilisent, leur nombre de cylindres, leur longueur, leur puissance et leur prix. Le carburant est un attribut de type catégoriel, les valeurs possibles sur cet attribut faisant partie d'un ensemble fini prédéterminé de catégories (diesel, essence, gpl et éthanol dans notre cas). Les autres attributs sont de type numérique, avec chacun son intervalle de définition (l'ensemble des réels compris entre 156 et 190 pour la longueur par exemple). La case en gras signifie ainsi que la troisième voiture utilise du diesel comme carburant.

Notons dès à présent qu'il existe différentes terminologies pour décrire les éléments de telles bases de données tabulaires. Le plus souvent, les lignes du tableau sont appelées les exemples, et les colonnes les attributs. On peut également considérer qu'il s'agit d'*objets* décrits par des valeurs sur plusieurs *dimensions*, ou bien de *points* décrits par leurs *coordonnées*. Enfin, une autre vue possible utilisée en statistiques est de considérer qu'il s'agit d'*individus* décrits par certaines *caractéristiques*, aussi appelées *variables*.

Notons aussi que dans certains cas, les données peuvent être *bruitées*, c'est-à-dire que certaines de leurs valeurs sont aberrantes. Il s'agit le plus souvent de valeurs issues de capteurs déficients, ou bien d'erreurs humaines dans leur manipulation. Il peut aussi arriver que certaines valeurs ne soient pas renseignées. On parle dans ce cas de *données manquantes*. Le tableau 1.2 présente un exemple de données dans lesquelles deux valeurs manquantes et une valeur aberrante (en gras) sont présentes.

C'est cette représentation par attribut-valeur qui est le plus souvent utilisée pour

identifiant	carburant	cylindres	longueur	puissance	prix
1	gpl	8	186	6000	16000
2	essence	4	170	5800	9000
3	**diesel**	6	172	5500	12000
4	diesel	4	156	5200	6500
5	essence	12	190	5500	19000
6	essence	4	175	5800	9500
7	éthanol	4	158	6000	8000
8	diesel	6	188	5200	18000
9	essence	4	168	5000	7500
10	diesel	6	170	6000	10500

TAB. 1.1 – Données stockées sous forme tabulaire.

identifiant	carburant	cylindres	longueur	puissance	prix
1	gpl	8	186	6000	16000
2	essence	4	170	5800	9000
3	diesel	6	172	**?**	12000
4	diesel	4	156	5200	6500
5	essence	12	190	5500	19000
6	**?**	4	175	5800	9500
7	éthanol	4	158	6000	8000
8	diesel	6	188	5200	18000
9	essence	4	**1680**	5000	7500
10	diesel	6	170	6000	10500

TAB. 1.2 – Données bruitées et contenant des valeurs manquantes.

décrire un ensemble de données, mais d'autres représentations plus complexes peuvent également être rencontrées. Par exemple, lorsque les données sont *symboliques* [Bock and Diday, 2000], alors les valeurs associées aux objets peuvent être des intervalles de définition sur un attribut numérique, ou bien un ensemble de modalités possibles sur un attribut catégoriel. De manière plus générale, on peut représenter les valeurs des objets par des distributions de probabilités sur les attributs considérés. Par ailleurs, certaines données peuvent être représentées par des *arbres*. C'est le cas par exemple de corpus de documents semi-structurés ou de documents étiquetés syntaxiquement. Certaines données sont fournies sous forme de *graphes*. Il est également possible qu'elles soient représentées par des *courbes*, indiquant par exemple l'évolution temporelle de certaines mesures. On parle dans ce cas de *données continues*, par opposition aux *données discrètes*. Enfin, lorsque l'on souhaite représenter des interactions complexes entre différents composants, comme en physique pour décrire les matériaux ou en chimie pour décrire les molécules, alors c'est souvent la *logique du premier ordre* qui est utilisée.

Dans les travaux que nous avons menés, nous nous sommes principalement intéressé au cas le plus souvent rencontré : celui des données représentées sous forme attribut-valeur et où les attributs peuvent être de nature numérique ou catégorielle. Dans un deuxième temps, nous nous sommes intéressé au cas des données représentées par des arbres.

1.3 L'apprentissage supervisé

L'objectif général de la *classification* est d'être capable d'étiqueter des données en leur associant une classe. Si les classes possibles sont connues et si les exemples sont fournis avec l'étiquette de leur classe, on parle d'*apprentissage supervisé* ou d'*analyse discriminante*. Dans ce cas, il s'agit alors d'utiliser les exemples fournis déjà classés pour apprendre un modèle qui permette ensuite d'associer à tout nouvel exemple rencontré sa classe la plus adaptée.

Un exemple d'application de l'apprentissage supervisé concerne la médecine : étant donnés les résultats d'analyse d'un patient, et la connaissance de l'état d'autres patients pour lesquels les mêmes analyses ont été menées, il est possible d'évaluer le risque de maladie de ce nouveau patient en fonction de la similarité de ses analyses avec celles des autres patients.

Dans le cas de notre exemple concernant les voitures, il peut s'agir par exemple de déterminer si une nouvelle voiture rencontrée fait partie de la classe des citadines, des voitures intermédiaires ou des voitures confortables, en se basant sur ses caractéristiques, et sur la classe connue d'autres voitures déjà rencontrées.

Le tableau 1.3 montre ainsi l'adaptation du tableau 1.1 dans le cadre de l'apprentissage supervisé. À chaque exemple initial est associée la classe de voiture à laquelle il appartient. L'objectif est alors d'être capable d'estimer la classe la plus appropriée à tout nouvel exemple rencontré. Par exemple, observant que plus les voitures sont longues, plus elles sont confortables, on pourrait extrapoler et répondre que la nouvelle voiture appartient à la classe intermédiaire.

identifiant	carburant	cylindres	longueur	puissance	classe
1	gpl	8	186	6000	confort
2	essence	4	170	5800	intermédiaire
3	diesel	6	172	5500	intermédiaire
4	diesel	4	156	5200	citadine
5	essence	12	190	5500	confort
6	essence	4	175	5800	intermédiaire
7	éthanol	4	158	6000	citadine
8	diesel	6	188	5200	confort
9	essence	4	168	5000	intermédiaire
10	diesel	6	170	6000	?

TAB. 1.3 – Données utilisées en apprentissage supervisé.

Par ailleurs, plutôt que d'associer à chaque exemple une classe unique, il est parfois utile de leur associer une probabilité d'appartenance à chacune des classes. Le tableau 1.4 présente le problème dans ce cadre. Ainsi, la première voiture n'est par exemple plus simplement classée comme confortable : elle a dans ce cas une probabilité de 0.8 d'être de la classe confort, et une probabilité de 0.2 d'être de la classe intermédiaire.

identifiant	carburant	cylindres	longueur	puissance	citadine	intermédiaire	confort
1	gpl	8	186	6000	0	0.2	0.8
2	essence	4	170	5800	0.1	0.9	0
3	diesel	6	172	5500	0.1	0.8	0.1
4	diesel	4	156	5200	0.9	0.1	0
5	essence	12	190	5500	0	0	1
6	essence	4	175	5800	0.1	0.8	0.1
7	éthanol	4	158	6000	0.8	0.2	0
8	diesel	6	188	5200	0	0.1	0.9
9	essence	4	168	5000	0.2	0.8	0
10	diesel	6	170	6000	?	?	?

TAB. 1.4 – Données utilisées en apprentissage supervisé probabiliste.

Enfin, lorsque la classe des exemples n'est pas catégorielle mais numérique, on parle de *régression*. Le tableau 1.5 adapte notre exemple sur les voitures dans ce cas. L'objectif dans ce cadre n'est alors plus simplement d'associer l'une des trois classes prédéterminées à la nouvelle voiture rencontrée, mais d'approcher au mieux le prix de cette voiture.

1.4 L'apprentissage non supervisé

A contrario, si seuls des exemples sans étiquette sont disponibles, et si les classes et leur nombre sont inconnus, on parle d'*apprentissage non supervisé*, ou *clustering*.

identifiant	carburant	cylindres	longueur	puissance	prix
1	gpl	8	186	6000	16000
2	essence	4	170	5800	9000
3	diesel	6	172	5500	12000
4	diesel	4	156	5200	6500
5	essence	12	190	5500	19000
6	essence	4	175	5800	9500
7	éthanol	4	158	6000	8000
8	diesel	6	188	5200	18000
9	essence	4	168	5000	7500
10	diesel	6	170	6000	?

TAB. 1.5 – Données utilisées en régression.

Dans ce cas, l'apprentissage se ramène alors à cibler les groupes homogènes d'exemples existant dans les données, c'est-à-dire à identifier des groupes tels que les exemples les plus similaires appartiennent au même groupe, et que les exemples les plus différents soient séparés dans différents groupes, la notion de similarité étant le plus souvent ramenée à une fonction de distance entre paires d'exemples. Autrement dit, il s'agit à ce niveau de rechercher la distribution sous-jacente des exemples dans leur espace de description.

En médecine, il peut par exemple être intéressant de détecter, parmi un ensemble de patients atteints d'une maladie donnée, les différents groupes de malades reflétant différentes causes possibles de la maladie. Comme le montre le tableau 1.6, il n'y a dès lors dans ce cas plus aucune information de classe associée aux exemples, ce qui revient finalement à travailler directement sur le tableau 1.1 initial.

identifiant	carburant	cylindres	longueur	puissance	prix	classe
1	gpl	8	186	6000	16000	?
2	essence	4	170	5800	9000	?
3	diesel	6	172	5500	12000	?
4	diesel	4	156	5200	6500	?
5	essence	12	190	5500	19000	?
6	essence	4	175	5800	9500	?
7	éthanol	4	158	6000	8000	?
8	diesel	6	188	5200	18000	?
9	essence	4	168	5000	7500	?
10	diesel	6	170	6000	10500	?

TAB. 1.6 – Données utilisées en apprentissage non supervisé.

La figure 1.1 montre par exemple une façon de regrouper un ensemble de voitures en fonction de leur longueur et de leur prix. La découverte de tels groupes permet ainsi de

mettre en avant le fait que le prix des voitures augmente significativement en fonction de leur longueur.

De même que pour l'apprentissage supervisé, on peut soit considérer que chaque exemple ne peut appartenir qu'à un seul groupe, soit considérer que chaque exemple a une probabilité donnée d'appartenir à chacun des groupes créés. Dans le premier cas, on parle de *hard clustering*, et dans le second de *soft clustering*.

1.5 L'apprentissage semi-supervisé

Il existe également des problèmes intermédiaires, où seules certaines données sont étiquetées. On parle alors d'*apprentissage semi-supervisé*. Le tableau 1.7 montre l'adaptation de notre exemple dans ce cadre.

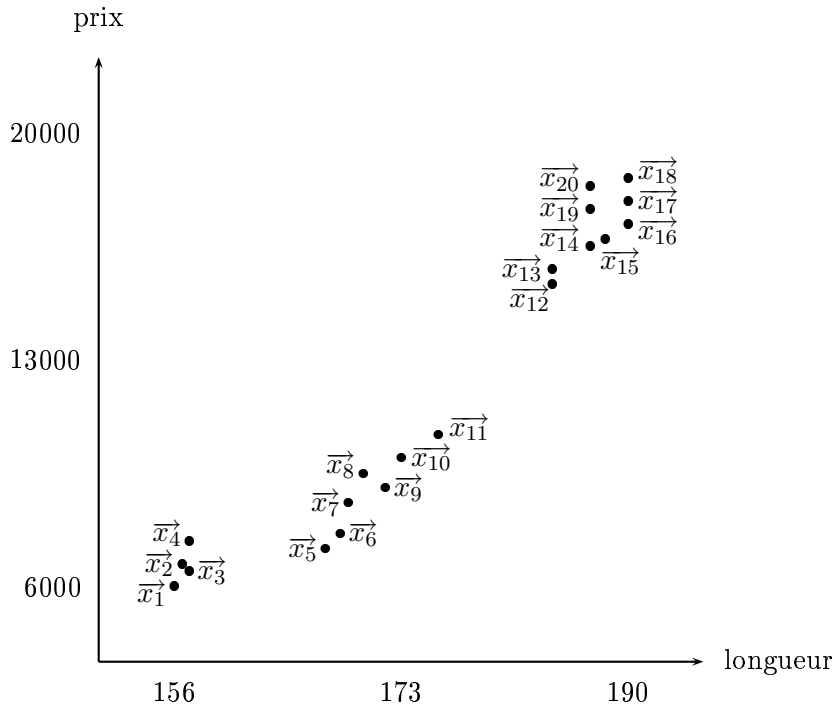
identifiant	carburant	cylindres	longueur	puissance	classe
1	gpl	8	186	6000	confort
2	essence	4	170	5800	intermédiaire
3	diesel	6	172	5500	intermédiaire
4	diesel	4	156	5200	citadine
5	essence	12	190	5500	confort
6	essence	4	175	5800	?
7	éthanol	4	158	6000	?
8	diesel	6	188	5200	?
9	essence	4	168	5000	?
10	diesel	6	170	6000	?

TAB. 1.7 – Données utilisées en apprentissage semi-supervisé.

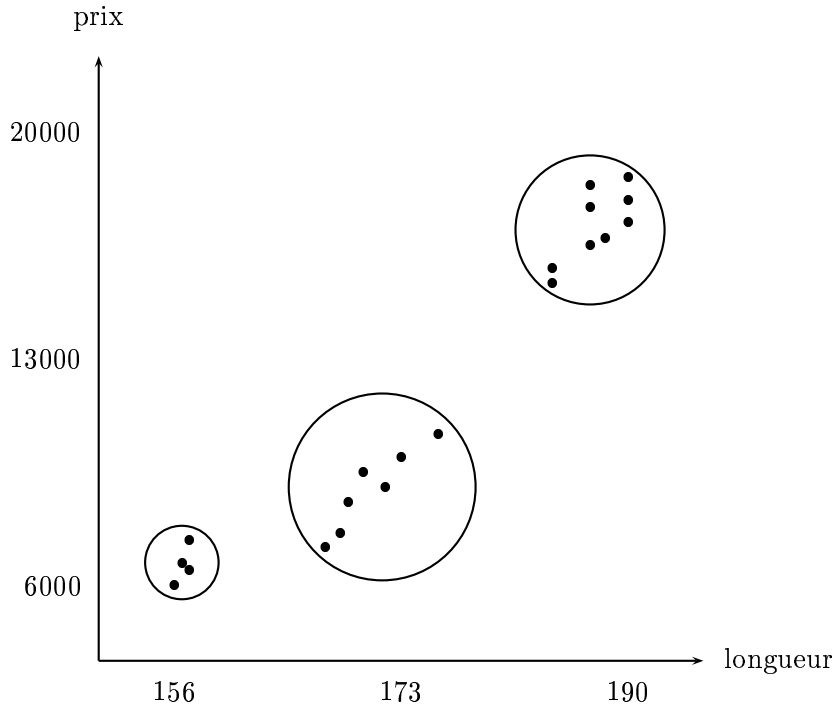
Dans ce cas, l'objectif est, de même qu'en apprentissage supervisé, de proposer un modèle permettant de représenter l'appartenance des objets à différentes classes prédéterminées, mais cette fois-ci en se servant également des exemples non étiquetés qui, comme en apprentissage non supervisé, sont susceptibles d'apporter de l'information quant à la distribution sous-jacente des exemples dans leur espace de description [Nigam et al., 2000].

Typiquement, ce type de problème se présente lorsqu'évaluer la classe d'un exemple est coûteux. C'est par exemple le cas lorsque l'on souhaite déterminer si un patient donné est atteint ou non d'une maladie qu'on ne peut détecter que par des moyens modernes honéreux.

De nouveau, il est également possible d'associer à chaque exemple une probabilité d'appartenance à chacune des classes. Le tableau 1.8 présente le problème dans ce cadre.



(a) Un exemple d'ensemble de données à classer.



(b) Et un regroupement possible de ces données.

FIG. 1.1 – Exemple d'apprentissage non supervisé.

identifiant	carburant	cylindres	longueur	puissance	citadine	intermédiaire	confort
1	gpl	8	186	6000	0	0.2	0.8
2	essence	4	170	5800	0.1	0.9	0
3	diesel	6	172	5500	0.1	0.8	0.1
4	diesel	4	156	5200	0.9	0.1	0
5	essence	12	190	5500	0	0	1
6	essence	4	175	5800	?	?	?
7	éthanol	4	158	6000	?	?	?
8	diesel	6	188	5200	?	?	?
9	essence	4	168	5000	?	?	?
10	diesel	6	170	6000	?	?	?

TAB. 1.8 – Données utilisées en apprentissage semi-supervisé probabiliste.

1.6 L'apprentissage partiellement supervisé

Enfin, lorsque l'étiquetage des données est seulement partiel, comme dans le cas du tableau 1.9, on parle d'*apprentissage partiellement supervisé* [Ambroise and Govaert, 2000].

identifiant	carburant	cylindres	longueur	puissance	citadine	intermédiaire	confort
1	gpl	8	186	6000	0	0	1
2	essence	4	170	5800	0	1	0
3	diesel	6	172	5500	0	1	0
4	diesel	4	156	5200	1	0	0
5	essence	12	190	5500	0	0	1
6	essence	4	175	5800	?	?	?
7	éthanol	4	158	6000	?	?	0
8	diesel	6	188	5200	0	?	?
9	essence	4	168	5000	?	?	0
10	diesel	6	170	6000	?	?	?

TAB. 1.9 – Données utilisées en apprentissage partiellement supervisé.

De cette manière, on ajoute dans la base de données les informations partielles concernant les classes associées aux exemples. Dans notre exemple, on précise ainsi que la septième voiture est soit de la classe citadine, soit de la classe intermédiaire, mais qu'elle n'est pas de la classe confort puisque sa probabilité d'appartenance à cette classe est nulle. Dans le domaine médical, ce type d'information partielle correspond par exemple au fait qu'un médecin soit certain qu'un patient donné n'est pas atteint de la maladie A, mais qu'il hésite entre la maladie B ou la maladie C.

Le tableau 1.10 présente finalement le cas où les informations sur les classes des exemples sont partielles et où une probabilité d'appartenance à chacune des classes est associée à chaque exemple.

identifiant	carburant	cylindres	longueur	puissance	citadine	intermédiaire	confort
1	gpl	8	186	6000	0	0.2	0.8
2	essence	4	170	5800	0.1	0.9	0
3	diesel	6	172	5500	0.1	0.8	0.1
4	diesel	4	156	5200	0.9	0.1	0
5	essence	12	190	5500	0	0	1
6	essence	4	175	5800	?	?	?
7	éthanol	4	158	6000	?	?	0
8	diesel	6	188	5200	0	?	?
9	essence	4	168	5000	?	?	0
10	diesel	6	170	6000	?	?	?

TAB. 1.10 – Données utilisées en apprentissage partiellement supervisé probabiliste.

1.7 Problématiques générales

L'efficacité des méthodes mises en œuvre dépend fortement des caractéristiques des bases de données étudiées. Il y a ainsi quatre éléments principaux à prendre en compte :

1. le nombre d'exemples considérés,
2. le nombre et le type des attributs décrivant ces exemples,
3. la présence de données manquantes (les valeurs non renseignées),
4. et le bruit qui peut exister dans les données (les valeurs aberrantes).

Par ailleurs, une attention particulière doit être portée sur certains points lors de la mise en œuvre de méthodes d'apprentissage :

1. la minimisation des connaissances a priori requises de la part de l'utilisateur,
2. la complexité du modèle utilisé pour représenter les données,
3. et éventuellement la présentation des résultats sous forme compréhensible pour l'utilisateur.

Tout d'abord, l'efficacité des méthodes est fortement liée au nombre d'exemples qui leur sont fournis en entrée. Si ce nombre est trop faible, il peut alors être très difficile d'en extraire de l'information pertinente car les hypothèses envisagées peuvent alors être très nombreuses et ont un faible *support*, c'est-à-dire que peu d'exemples vérifient ces hypothèses, ce qui met en doute leur validité. Au contraire, si ce nombre est trop élevé, alors des problèmes d'échelles peuvent se poser. Il faut alors être capable de gérer les problèmes de stockage en mémoire de grandes quantités d'informations, ainsi que du temps d'exécution de la méthode proposée.

De la même façon, des données décrites par un grand nombre d'attributs nécessitent souvent des méthodes spécifiques capables de prendre en compte de telles quantités importantes d'attributs. La nature très variée des attributs doit également être considérée lors de la mise en œuvre de méthodes d'apprentissage. En effet, les attributs numériques et les attributs catégoriels peuvent rarement être traités de la même manière. Par exemple, la notion de distance entre deux exemples n'est pas la même selon que les

attributs qui les caractérisent sont numériques ou catégoriels, car s'il est assez direct de postuler qu'une voiture coûtant 10000 euros est plus proche d'une voiture coûtant 9000 euros que d'une voiture coûtant 18000 euros, il est beaucoup plus problématique d'évaluer la proximité entre trois voitures dont l'une est rouge, l'autre bleue et la troisième jaune.

Par ailleurs, un autre élément important à prendre en compte lors de la mise en œuvre d'une méthode d'apprentissage est la possibilité que les valeurs de certaines données sur certains attributs ne soient pas renseignées, ainsi que la possible présence de données bruitées, c'est-à-dire d'exemples qui ne suivent pas la distribution générale des exemples sur leur espace de description.

Un autre objectif important consiste à minimiser les connaissances a priori requises de la part de l'utilisateur, afin que la méthode soit la plus générique possible, et parce qu'il est souvent très difficile pour les utilisateurs de fournir de telles connaissances. En apprentissage non supervisé, il est par exemple souvent demandé à l'utilisateur de spécifier le nombre de groupes présents dans les données, alors que cette information est peu souvent disponible en pratique. Nous verrons dans la suite que ce point a été très présent lors de nos différents travaux de recherche.

D'autre part, une attention particulière doit être portée sur la complexité du modèle proposé pour représenter les données. On retrouve à ce niveau le compromis classique entre *spécificité* et *généralisation* du modèle. En effet, si un modèle trop complexe est proposé, alors il risque d'être trop spécifique aux données d'apprentissage, et donc être difficilement généralisable à de nouvelles données rencontrées. On parle dans ce cas de *sur-apprentissage*. Au contraire, si un modèle trop simple est proposé, alors il risque de ne pas être capable de cibler certains concepts spécifiques dans les données car les concepts ciblés sont trop généraux. On parle dans ce cas de *sous-apprentissage*.

Enfin, un dernier point qui se révèle très utile dans certains cas, et qui constitue également une part importante de nos travaux de recherche, étant donné le cadre dans lequel ils ont été menés, est celui de la présentation de résultats qui soient facilement interprétables par l'utilisateur.

1.8 Cadre de la recherche

La société Pertinence[1] propose aujourd'hui un logiciel très efficace permettant d'extraire de manière compréhensible de la connaissance à partir de bases de données étiquetées hétérogènes, issues pour la plupart de processus industriels complexes, puis de gérer de façon pertinente cette connaissance dans le but d'améliorer au mieux ces processus et leur maintenance.

Autrement dit, Pertinence a développé une méthode efficace d'apprentissage supervisé à partir de bases de données pouvant contenir différents types d'attributs, et qui fournit en sortie un résultat compréhensible sous forme de règles définies par un minimum d'attributs parmi les plus pertinents, ainsi qu'une méthode permettant d'utiliser au mieux ces règles produites.

[1] http ://www.pertinence.com

On appelle ici *règle* une conjonction de tests définissant l'appartenance d'un ensemble d'exemples à une classe. Ces tests sont définis sur un sous-ensemble des attributs initiaux de description. Un test sur un attribut numérique définit l'appartenance de la valeur d'un exemple à un intervalle de définition donné, et un test sur un attribut catégoriel à l'appartenance de la valeur de l'exemple à une ou plusieurs catégories prédéfinies. La figure 1.2 montre une telle règle, illustrant la facilité d'interprétation du résultat produit.

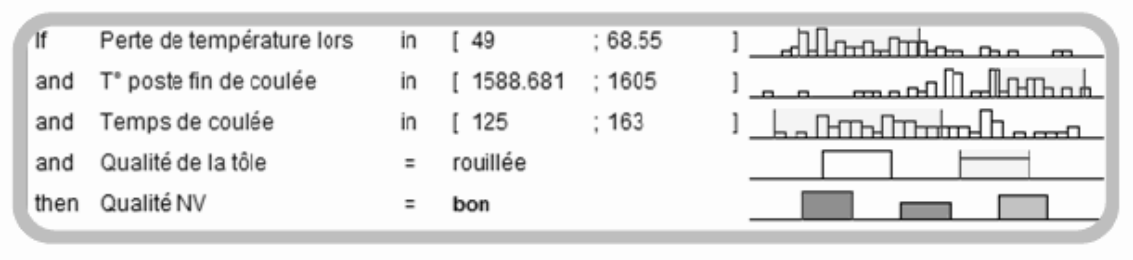


FIG. 1.2 – Exemple de règle produite par Pertinence Rule Maker.

Pertinence Rule Maker ^{TM} ne nécessite aucune connaissance a priori de la part de l'utilisateur et est capable de faire face aux données réelles pouvant contenir des données manquantes, bruitées, et caractérisées par de nombreux attributs de types différents et parmi lesquels plusieurs peuvent ne pas être pertinents.

Le cadre de notre recherche se situe dans l'extension de cette méthode efficace au cas où toutes les informations de classes ne sont pas disponibles. Autrement dit, nous cherchons à étendre la capacité du logiciel aux méthodes non supervisées, semi-supervisées et partiellement supervisées, tout en conservant ses atouts actuels, à savoir :

- faire face à des données réelles
- pouvant contenir des données manquantes et du bruit,
- caractérisées par de nombreux attributs de types différents,
- et parmi lesquels plusieurs peuvent ne pas être pertinents,
- et produire un résultat facilement interprétable par l'utilisateur
- sans nécessiter de connaissance a priori de sa part.

Nous verrons alors dans la littérature les solutions proposées pour faire face à ces différentes problématiques, puis nous présenterons nos contributions dans ce domaine. Trois thématiques principales se sont dégagées :

1. la *contextualisation* fait référence au problème de la prise en compte du contexte dans lequel les groupes sont formés, c'est-à-dire le fait que certains groupes d'exemples peuvent être caractérisés par différents sous-ensembles des attributs initiaux ;
2. la *visualisation* fait référence à la problématique de la présentation des résultats de manière compréhensible à l'utilisateur ;
3. et l'*évaluation* est un problème très particulier dans le cadre de l'apprentissage non supervisé car l'intérêt même d'un regroupement est difficile à définir car subjectif par nature.

Nous verrons alors l'intérêt d'utiliser des *modèles probabilistes* pour faire face au problème de la contextualisation ainsi qu'à celui de la mise en œuvre d'une méthode capable de faire face aux données manquantes, au bruit présent dans les données et à différents types d'attributs décrivant les exemples, et ne requiérant pas de connaissance a priori de la part de l'utilisateur. Nous verrons l'intérêt d'utiliser des *règles* et leur *projection* en deux dimensions afin de présenter les résultats de manière compréhensible à l'utilisateur. Et nous verrons également l'intérêt de *replacer le clustering dans son contexte d'utilisation* pour son évaluation.

Nous commençons donc naturellement par présenter l'état de l'art en clustering dans la partie I. Nos contributions algorithmiques dans ce domaine, ainsi que les expérimentations que nous avons menées dans ce cadre, sont ensuite présentées dans la partie II. Puis la partie III est consacrée à nos contributions dans le domaine de l'évaluation des résultats d'algorithmes de clustering, ou de la comparaison de tels algorithmes, qui constitue une problématique ouverte importante dans ce cadre. Enfin, les conclusions et perspectives ouvertes par nos travaux sont présentées dans la partie IV.

Plusieurs annexes en partie V détaillent finalement plusieurs points évoqués, permettant ainsi une lecture plus fluide de la thèse pour le lecteur ne souhaitant pas rentrer dans les détails de toutes les techniques existantes. Un bilan est également présenté à la fin de chaque section pour offrir un troisième axe de lecture *en diagonal* pour le lecteur souhaitant connaître rapidement les principales contributions de cette thèse.

Première partie

État de l'art en clustering

Le clustering est une problématique de recherche étudiée depuis de nombreuses années dans différentes communautés : machine learning, data mining, pattern recognition, statistiques, etc.

Son objectif, très général, consiste à séparer un ensemble d'objets en différents groupes (ou *clusters*) en fonction d'une certaine notion de similarité. Les objets qui sont considérés comme similaires sont ainsi associés au même cluster alors que ceux qui sont considérés comme différents sont associés à des clusters distincts.

Selon l'application envisagée pour le clustering, les considérations ne sont pas les mêmes, et la définition même de l'intérêt d'une solution peut varier. Par exemple, lorsque l'objectif est de réduire la taille d'un jeu de données pour un traitement plus efficace, alors une solution est jugée optimale lorsqu'elle minimise la perte d'information liée à une telle compression des données. Si le clustering est utilisé sur des bases de données de clients dans le but de faire émerger des groupes de clients aux comportements différents, alors le critère de qualité d'un résultat est principalement basé sur une séparation importante des groupes formés. Dans d'autres cas au contraire, considérer des clusters qui se chevauchent peut se révéler plus intéressant. Enfin, il est également possible de définir l'intérêt d'un clustering par une notion de densité des clusters produits. Les méthodes basées sur de tels critères sont alors en général mieux adaptées que d'autres pour cibler des clusters de forme quelconque, alors que ces dernières seront plus efficaces si l'objectif est de fournir une représentation compréhensible des clusters identifiés.

Il est ainsi admis dans la communauté travaillant sur le clustering qu'aucun critère ni aucune méthode ne sont intrinsèquement meilleurs que d'autres sur l'ensemble des problématiques et des problèmes envisageables. Par contre, certains critères et certaines méthodes sont plus appropriés que d'autres dans certains champs d'application et face à certains problèmes.

Dans cette partie, nous ne prétendons pas fournir une liste exhaustive de l'ensemble des notions et méthodes existant dans le cadre du clustering, mais plutôt de donner un aperçu général des différentes problématiques rencontrées dans ce cadre, ainsi que des méthodes les plus utilisées pour y faire face.

Nous présentons tout d'abord dans le chapitre 2 une vue générale du problème. Puis le chapitre 3 est consacré à la présentation des méthodes de clustering les plus classiques. Pour chacune de ces méthodes, nous donnons un aperçu général de ses caractéristiques principales, de ses forces et de ses faiblesses. Ainsi, selon l'application envisagée pour le clustering, l'utilisateur pourra rechercher la méthode la plus appropriée en fonction de l'ensemble de ses caractéristiques.

De plus amples détails sont fournis par la suite en annexes A.

Chapitre 2

Vue générale du problème

Ce chapitre propose une vue générale du problème du clustering. Nous présentons tout d'abord les différentes applications possibles du clustering. Puis nous discutons des différentes notions qui sont utilisées pour définir la similarité entre objets, qui constitue la base de toute méthode de clustering. Nous introduisons ensuite le problème de la prise en compte du *contexte* dans lequel les clusters sont créés. La section suivante présente de façon très succinte les principales méthodes de clustering existantes, ainsi qu'une liste des caractéristiques principales qui peuvent leur être associées. Enfin, nous présentons les approches existantes permettant d'évaluer les résultats d'algorithmes de clustering, avant d'introduire quelques notations qui seront utilisées tout au long de la thèse. De plus amples détails sur les principales méthodes de clustering existantes sont ensuite présentés dans le chapitre suivant.

2.1 Applications du clustering

Il existe de nombreuses applications possibles au clustering, que l'on peut classer en trois groupes principaux [Berkhin, 2002] :

1. la *segmentation*,
2. la *classification*,
3. et l'*extraction de connaissances*.

La *segmentation* d'une base de données a pour objectif principal de réduire la taille de l'ensemble des données considérées afin de favoriser leur traitement. On parle dans ce cas de *condensation* ou de *compression* des données, celles-ci pouvant alors être considérées collectivement. Une telle technique est par exemple utile en segmentation d'images, afin d'identifier les différentes zones homogènes de l'espace décrit (les champs, maisons, routes, fleuves, etc.). De manière plus générale, une telle méthode est utile pour segmenter l'espace dans des bases de données spatiales. En particulier, identifier différents *sous-espaces* homogènes pouvant exister dans des bases de données OLAP peut aider à l'indexation efficace de telles bases. Enfin, la segmentation est parfois utilisée pour *discrétiser* une base de données, c'est-à-dire transformer la description

complexe des objets par un unique attribut caractérisant leur appartenance à une classe identifiée automatiquement.

La *classification* concerne typiquement l'identification de *sous-populations* ayant des caractéristiques proches dans les bases de données clients. Dans ce cas, l'objectif principal est d'établir des profils de clients aux comportements similaires, afin de comprendre leurs attentes et leurs besoins en fonction de leur profil. De telles informations permettent alors d'adapter les actions marketing envers les clients en fonction de leur appartenance à une population donnée. Les applications principales de telles techniques concernent donc la *Gestion de la Relation Client*, que ce soit dans la grande distribution, les banques, les assurances, les fournisseurs d'énergie, les opérateurs téléphoniques, les transports, les médias, les campagnes de mailings, les navigateurs web, etc. Cela peut également être appliqué aux cas des journaux en ligne ou des forums par exemple.

Enfin, contrairement aux deux méthodes précédentes, l'utilisation du clustering pour l'*extraction de connaissances* n'a pas d'objectif préétabli. Il s'agit dans ce cas d'aider à la compréhension de la structure des données en les organisant en groupes homogènes et en faisant ainsi émerger certains *sous-concepts* dans les données. On parle dans ce cas de *génération d'hypothèses* ou de *modélisation prédictive*, par l'inférence de règles caractérisant les données, et la suggestion de modèles sous-jacents à ces données. L'intérêt de telles techniques est par exemple avéré pour effectuer des diagnostics médicaux sur des bases de données de patients, en ciblant par exemple différents groupes de patients atteints d'une maladie donnée, reflétant ainsi différentes causes possibles pour la maladie. De même, cette technique peut être utilisée en analyse du web, en analyse des données textuelles, en analyse de ventes, en bio-informatique, en analyse génétique, etc.

2.2 Notions de similarité

L'objectif du clustering est de regrouper un ensemble de données de la manière la plus naturelle possible. Cette volonté de *regrouper naturellement* est bien sûr ambiguë et le plus souvent formalisée par l'objectif de définir des groupes d'objets tels que la similarité entre objets d'un même groupe soit maximale et que la similarité entre objets de groupes différents soit minimale.

Le problème est alors de définir cette notion de similarité entre objets. Typiquement, la similarité entre objets est estimée par une fonction calculant la distance entre ces objets. Une fois cette fonction distance définie, la tâche de clustering consiste alors à réduire au maximum la distance entre membres d'un même cluster tout en augmentant au maximum la distance entre clusters.

Cette vision de l'apprentissage non supervisé contraint donc à disposer d'une distance définie sur le langage de description des objets. Ainsi, deux objets proches selon cette distance seront considérés comme similaires, et au contraire, deux objets séparés par une large distance seront considérés comme différents.

Le choix de cette mesure de distance entre objets est très important. Malheureusement, trop souvent, il s'agit d'un choix arbitraire, sensible à la représentation des objets, et qui traite tous les attributs de la même manière.

Une solution pour pallier à cette limitation est celle de la prise en compte de la connaissance d'un expert, qui identifiera certains attributs, considérés comme plus pertinents que d'autres pour le problème considéré, et leur attribuera un poids plus important lors du calcul des distances entre objets.

Cependant, cette solution devient très difficile à mettre en œuvre lorsque le nombre d'attributs décrivant les données est trop grand, ou qu'il n'y a pas d'expert humain. Différentes méthodes ont donc été proposées pour effectuer une sélection ou une extraction automatique d'attributs préalable à l'apprentissage. Les objets sont alors projetés dans un espace de taille inférieure à l'espace initial de description, mais tous restent décrits par le même ensemble d'attributs.

2.3 Prise en compte du contexte

De nouvelles difficultés apparaissent lorsque l'on tente de prendre en compte le fait que certains attributs composant les données ont plus ou moins d'importance dans la construction de certains clusters.

Ce problème a des échos en apprentissage supervisé. Il s'agit du fait qu'un concept puisse être disjonctif [Torre, 1999]. Par exemple, dans le cas où l'on cherche à déterminer quelles sont les personnes qui ont un risque cardio-vasculaire, on s'aperçoit que chez les femmes, le risque est amené par le diabète, et que le taux de cholestérol, même élevé, n'a que peu d'importance. À l'inverse, chez les hommes, le taux de cholestérol sera déterminant pour l'évaluation du risque, alors que le diabète n'aura pas à être pris en compte.

Dans le cas de l'apprentissage non supervisé, la même problématique se pose car certains attributs peuvent être discriminants pour la formation d'un certain cluster, alors que ces mêmes attributs peuvent s'avérer peu révélateurs pour la formation d'un autre cluster. Le *contexte* dans lequel seront placés les attributs va donc déterminer la pertinence de leur prise en compte pour le clustering. Autrement dit, tous les attributs ne sont pas forcément utiles et les attributs pertinents ne sont pas nécessairement les mêmes d'un cluster à l'autre.

Or, face à cette problématique, la simple utilisation de la notion de distance entre objets va échouer. En effet, la distance entre deux objets étant définie globalement, cette technique ne pourra prendre en compte le fait que cette distance peut varier selon le contexte, c'est-à-dire selon le cluster considéré. La figure 2.1 montre l'exemple d'un tel jeu de données dans lequel sont présents plusieurs clusters caractérisés par certains attributs qui leur sont propres.

Par exemple, si l'on considère le groupe formé par l'ensemble des objets représentés par des ronds, on remarque qu'il est caractérisé par de faibles valeurs sur la dimension X et des valeurs élevées sur la dimension Z ; par contre, les valeurs possibles de ces objets sur la dimension Y couvrant l'ensemble de l'intervalle de définition de Y, cette dimension n'aide pas à les différencier des autres objets. Le sous-espace de description de ce groupe est donc $X \times Z$. De même, le sous-espace de description du groupe d'objets représentés par des triangles est $Y \times Z$ car les valeurs de ces objets sur la dimension X

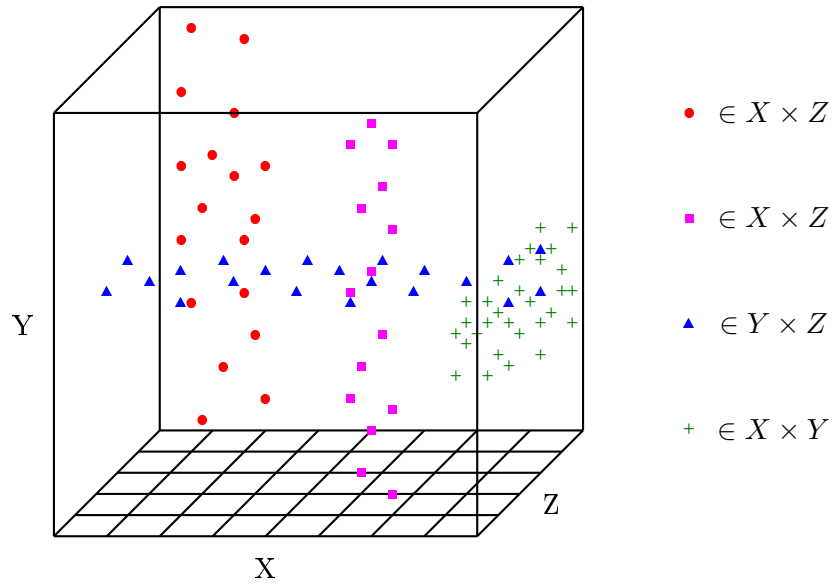


FIG. 2.1 – Exemple de quatre clusters définis dans des sous-espaces différents.

ne fournit aucune information sur leur appartenance à ce groupe.

Par cet exemple, on voit donc bien que pour un même ensemble d'objets, différents groupes peuvent être caractérisés en fonction de différents sous-ensembles des attributs initiaux. Or sans cette prise en compte du contexte dans lequel les clusters sont formés, ceux-ci ne peuvent être identifiés correctement car une notion de distance définie globalement sur l'espace complet de description des objets ne le permet pas.

2.4 Méthodes de clustering

On distingue classiquement deux grandes familles de méthodes en clustering : les méthodes *hiérarchiques* et les méthodes *par partition*. Dans le premier cas, une hiérarchie de clusters est formée de telle manière que plus on descend dans la hiérarchie, plus les clusters sont spécifiques à un certain nombre d'objets considérés comme similaires. Au contraire, dans le second cas, le résultat fourni est une partition de l'espace des objets, c'est-à-dire que chaque objet est associé à un unique cluster.

Dans le cas du clustering hiérarchique, deux grandes méthodes se distinguent :

1. les méthodes ascendantes, qui démarrent avec autant de clusters que d'objets puis fusionnent successivement l'ensemble des clusters selon un certain critère jusqu'à ce que tous les objets soient finalement regroupés dans un unique cluster stocké à la racine de la hiérarchie ;

2. et les méthodes descendantes, qui démarrent avec un unique cluster regroupant l'ensemble des objets, puis divisent successivement les clusters selon un certain critère jusqu'à ce que tous les objets se retrouvent dans des clusters différents stockés aux feuilles de la hiérarchie.

On retrouve donc à ce niveau la différence classique en apprentissage entre généralisation et spécialisation : les méthodes ascendantes démarrent avec une solution tout à fait spécifique aux données, qui est ensuite généralisée à chaque étape, alors que les méthodes descendantes démarrent avec une solution complètement générale, qui est ensuite spécialisée à chaque étape.

Dans le cas du clustering par partition, plusieurs méthodes se distinguent fortement :

- le *clustering statistique* est basé sur l'hypothèse que les données ont été générées en suivant une certaine loi de distribution, le but étant alors de trouver les paramètres de cette distribution, ainsi que les paramètres cachés déterminant l'appartenance des objets aux différentes composantes de cette loi ; le *clustering basé sur les K-moyennes*, méthode très souvent utilisée, en est un cas particulier ;
- le *clustering stochastique* consiste à parcourir l'espace des partitions possibles selon certaines heuristiques, et à sélectionner celle qui optimise un critère donné ;
- le *clustering basé sur la densité* a pour but d'identifier dans l'espace les zones de forte densité entourées par des zones de faible densité pour la formation des clusters ;
- comme son nom l'indique, le *clustering basé sur les grilles* utilise une grille pour partitionner l'espace de description des objets en différentes cellules, puis identifie les ensembles de cellules denses connectées pour former les clusters ;
- le *clustering basé sur les graphes* consiste à former le graphe connectant les objets entre eux et dont la somme des valeurs des arcs, correspondant aux distances entre les objets, est minimale, puis à supprimer les arcs de valeurs maximales pour former les clusters ;
- enfin, la base du *clustering spectral* consiste à projeter itérativement les objets dans des sous-espaces de variance maximum, puis à utiliser une méthode de partitionnement dans de tels sous-espaces pour séparer les données.

Différentes méthodes *hybrides* ont également été proposées qui mélangent les caractéristiques des méthodes hiérarchiques et des méthodes par partition, cherchant ainsi à bénéficier des atouts de chaque méthode.

Enfin, une autre famille de méthodes ayant pour objectif de s'attaquer à la problématique du contexte évoquée dans la section précédente est apparue relativement récemment. Portant le nom de méthodes de *subspace clustering* [Parsons et al., 2004], elles ont pour objectif de cibler simultanément les groupes d'objets présents dans les données ainsi que les sous-espaces spécifiques dans lesquels ces groupes sont définis.

Ces méthodes sont plus générales qu'une méthode de clustering qui utilise une phase initiale de sélection ou d'extraction d'attributs car les sous-espaces sont spécifiques à chacun des clusters formés. Elles permettent de faire face au problème de la *malédiction de la dimensionalité* (*curse of dimensionality*) dans les bases de données contenant de nombreux attributs. De plus, elles permettent d'obtenir une description réduite des

clusters puisque chaque cluster est décrit par un nombre restreint d'attributs qui lui sont spécifiques. Deux grandes familles de subspace clustering se distinguent :

1. les méthodes ascendantes sur les dimensions, considérant des sous-espaces de dimensionalité croissante et ciblant ensuite les groupes d'objets existant dans ces sous-espaces ;
2. et les méthodes descendantes sur les dimensions, utilisant différentes techniques permettant de cibler les sous-espaces spécifiques à certains groupes d'objets considérés.

Comme nous l'avons déjà évoqué, aucune méthode de clustering ne peut être considérée comme meilleure que toutes les autres sur l'ensemble des problèmes envisageables. On considère plutôt que certaines méthodes sont plus adaptées que d'autres dans certains cas. Afin d'aider les utilisateurs à faire leur choix parmi les nombreuses méthodes existantes en fonction de l'application ciblée, nous proposons ici un ensemble de critères qui peuvent leur être associées :

- tout d'abord, les connaissances a priori requises de la part de l'utilisateur sont un premier critère important à prendre en compte dans le choix de la méthode ; il s'agit souvent d'informations concernant le nombre de clusters recherchés, la distance minimale entre clusters disjoints, ou la densité minimale à l'intérieur des clusters ;
- on peut ensuite différencier les méthodes en fonction de leur façon de présenter les résultats ; on peut en particulier différencier les méthodes fournissant en sortie une hiérarchie de clusters des méthodes fournissant en sortie une partition de l'ensemble des objets ;
- la complexité des méthodes est évidemment également un critère important à considérer ; en particulier, il est admis que la complexité des méthodes doit être linéaire en fonction du nombre d'objets dans le cas de larges bases de données ; il faut dès lors éviter toute méthode basée sur le calcul des distances deux à deux entre objets, menant à une complexité quadratique en fonction du nombre d'objets ;
- il est également possible de distinguer les méthodes déterministes des méthodes stochastiques : avec les mêmes données en entrée, un algorithme déterministe exécutera toujours la même suite d'opérations et fournira donc toujours le même résultat alors qu'une méthode stochastique pourra donner des résultats différents car elle permet l'exécution d'opérations aléatoires ;
- une autre caractéristique qui peut se révéler importante pour les méthodes de clustering est leur capacité de traiter des données de façon incrémentale, c'est-à-dire en les intégrant au fur et à mesure de leur arrivée dans l'algorithme ; à l'inverse, une méthode non-incrémentale va considérer un ensemble de données fournies en entrée et sera exécutée sur cet ensemble de données, et si par la suite une nouvelle donnée devait être fournie en entrée de l'algorithme, celui-ci devrait être relancé à nouveau ;
- de la même façon, on peut différencier les méthodes *any-time*, capables de fournir un résultat intermédiaire, même sous-optimal, à n'importe quel moment du dé-

roulement de la méthode, des méthodes nécessitant l'accomplissement total de la méthode avant d'être capable de fournir un résultat ;
- on peut ensuite différencier les méthodes *hard* qui associent à chaque objet un unique cluster, des méthodes *soft* qui associent à chaque objet un degré variable d'appartenance à chacun des clusters formés ; notons d'ailleurs dès à présent qu'un clustering *soft* peut être converti en clustering *hard* en assignant chaque objet au cluster dont la mesure d'appartenance est la plus forte ;
- une autre distinction possible concerne la capacité des méthodes à prendre en compte ou non la problématique du contexte évoquée précédemment ; autrement dit, on distingue à ce niveau les méthodes de subspace clustering des méthodes de clustering traditionnel ;
- il peut également être utile pour une méthode d'être capable de gérer le bruit qui peut exister dans les données, c'est-à-dire la possible présence d'objets qui ne suivent pas la distribution générale des autres objets ;
- en particulier, il s'avère souvent utile d'être capable de faire face au problème de l'*effet de chaîne*, illustré par la figure 2.2 et qui fait que des clusters proches mais distincts peuvent être fusionnés s'il existe une chaîne d'objets qui les relie ;

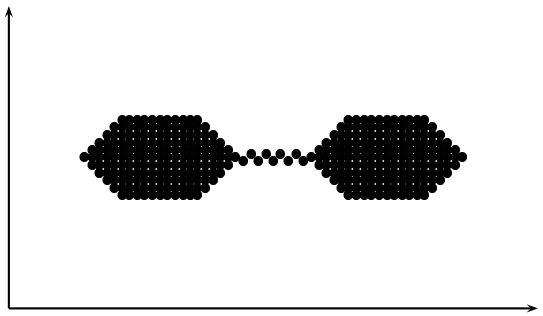


FIG. 2.2 – Problématique de l'effet de chaîne.

- enfin, un autre critère qui peut s'avérer important est le type de clusters que la méthode est capable de détecter ; en particulier, il est souvent utile d'être capable de détecter des clusters de tailles ou de densités variées, comme dans le cas de la figure 2.3 ;
- de même, il est dans certains cas utile qu'une méthode de clustering soit capable d'identifier des clusters de forme quelconque, et aussi éventuellement des clusters concentriques, c'est-à-dire inscrits les uns dans les autres, comme dans le cas de la figure 2.4.

Finalement, le tableau 2.1 résume l'ensemble de ces critères associés aux méthodes de clustering, avec leurs valeurs possibles. Étant donné que pour chacune des méthodes générales de clustering, plusieurs alternatives sont envisageables, nous ne présenterons que les caractéristiques associées à la méthode de base qui est décrite.

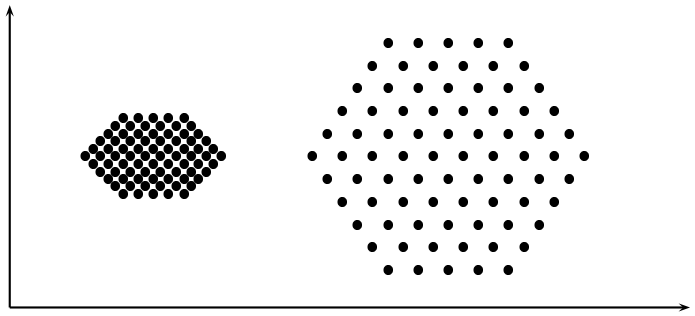


FIG. 2.3 – Problématique des clusters de tailles et de densités variées.

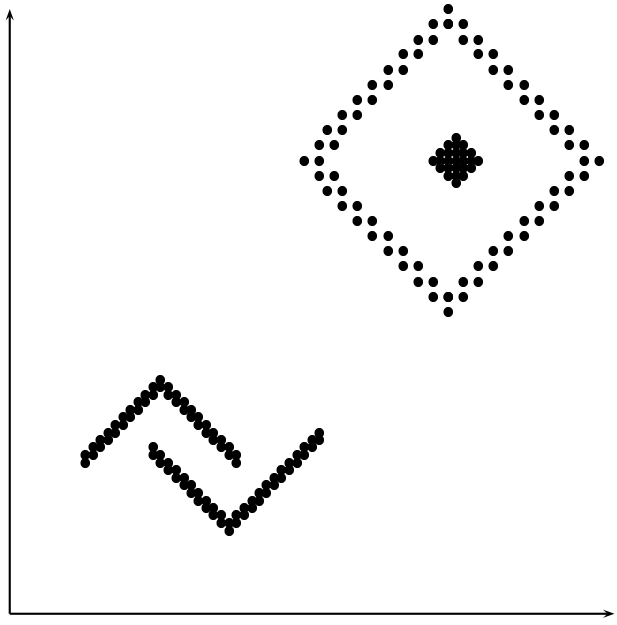


FIG. 2.4 – Problématique des clusters de formes variées et concentriques.

caractéristique	**valeurs possibles**
connaissances a priori	*
présentation des résultats	*
complexité	*
déterministe	oui / non
incrémental	oui / non
any-time	oui / non
hard	oui / non
prise en compte du contexte	oui / non
tolérance au bruit	oui / non
tolérance à l'effet de chaîne	oui / non
tolérance aux clusters de tailles variées	oui / non
tolérance aux clusters de densités variées	oui / non
tolérance aux clusters de forme quelconque	oui / non
tolérance aux clusters concentriques	oui / non

TAB. 2.1 – Caractéristiques associées aux méthodes de clustering.

2.5 Évaluation des résultats

En apprentissage supervisé comme en apprentissage non supervisé, l'évaluation des résultats d'une méthode donnée, de même que la comparaison de différentes méthodes, est une problématique importante. Mais si la *validation croisée* est une méthodologie largement utilisée pour l'évaluation en classification supervisée, bien que contestée dans [Bengio and Grandvalet, 2004], l'évaluation de la pertinence des groupes formés en classification non supervisée reste un problème ouvert. La difficulté vient principalement du fait que l'évaluation des résultats d'algorithmes de clustering est subjective par nature car il existe souvent différents regroupements pertinents possibles pour un même jeu de données.

En pratique, il existe quatre méthodes principales pour mesurer la qualité des résultats d'algorithmes de clustering, mais chacune souffre de différentes limitations.

1. Utiliser des données artificielles pour lesquelles le regroupement attendu est connu, et comparer les résultats obtenus aux résultats attendus. On parle dans ce cas d'*évaluation externe*. Mais les méthodes sont alors évaluées uniquement sur les distributions spécifiques correspondantes, et les résultats sur données artificielles ne peuvent pas être généralisés aux données réelles.
2. Utiliser des jeux de données étiquetés et observer si la méthode retrouve les classes initiales. Mais les classes d'un problème supervisé ne correspondent pas nécessairement aux groupes qu'il est pertinent d'identifier pour une méthode non supervisée, d'autres regroupements pouvant être plus appropriés.
3. Utiliser des critères numériques, tels l'inertie intra-cluster ou la séparation inter-clusters. On parle dans ce cas d'*évaluation interne*. Mais de tels critères ont une

part importante de subjectivité car ils utilisent des notions prédéfinies de ce qu'est un bon clustering. Par exemple, la séparation des clusters n'est pas toujours un bon critère à utiliser car parfois, des clusters qui se chevauchent peuvent être plus pertinents.

4. Utiliser un expert pour évaluer le sens d'un clustering donné dans un champ d'application spécifique. Mais s'il est possible à un expert de dire si un regroupement donné a du sens, il est beaucoup plus problématique de quantifier son intérêt ou de dire si un regroupement est meilleur qu'un autre. De plus, l'intérêt de la méthode ne peut être généralisé à différents types de jeux de données.

En pratique, on commence souvent par tester le comportement de la méthode de clustering évaluée face à des jeux de données artificiels, car cela permet d'observer son efficacité dans différentes situations contrôlées, et donc de vérifier ses propriétés. Après une telle phase préliminaire, il est ensuite nécessaire de confronter la méthode à des cas d'applications réels. Il faut alors disposer d'un expert du domaine étudié pour qu'il mette en avant la pertinence des résultats fournis. Cependant, même si une méthode obtient des résultats satisfaisants lors de ces deux étapes, son efficacité n'est pas garantie dans d'autres cas d'applications réels car, comme nous l'avons évoqué précédemment, les propriétés des données et les critères d'intérêt du clustering peuvent varier d'une application à l'autre.

2.6 Notations

Comme nous l'avons motivé dans la section 1.2, nous considérons dans la majeure partie de nos travaux que l'entrée d'un algorithme de clustering est une matrice D de taille $N \times M$, avec N le nombre d'objets de la base et M le nombre de dimensions formant l'espace de description S des objets. On notera $\overrightarrow{x_i} = (x_{i1}, ..., x_{iM})$, pour $i \in [1, N]$, le i^{eme} objet de la base, x_{id} correspondant à sa valeur sur la dimension d.

On supposera que toute dimension numérique est normalisée telle que l'intervalle de définition des objets soit $[0, 1]$. Ainsi, on évite qu'une dimension définie sur un intervalle initial plus large n'ait plus d'influence qu'une dimension définie sur un intervalle plus réduit. Cette phase sera cependant évitée si on se place dans le cadre d'une application où l'espace de description a un sens géométrique : l'espace euclidien dans le cadre de la segmentation de bases de données spatiales par exemple. Pour une dimension catégorielle d, on notera $Modalites_d$ l'ensemble des modalités possibles sur cette dimension, et $Frequences_d$ les fréquences de ces modalités sur l'ensemble de la base.

On notera K le nombre de clusters identifiés, C_k le k^{eme} de ces clusters, D_k l'ensemble des indices des objets associés au cluster C_k, N_k le nombre de ces objets associés, et P la partition créée composée de l'ensemble des clusters C_k pour $k \in [1, K]$. Par exemple, pour le cluster C_1 de la figure 2.5, on aura $D_1 = \{1, 2, 3, 4\}$ et $N_1 = 4$.

Enfin, pour les méthodes de subspace clustering qui associent à chaque cluster C_k un sous-ensemble des dimensions de S formant son sous-espace spécifique, on notera S_k le sous-ensemble de ces dimensions sélectionnées.

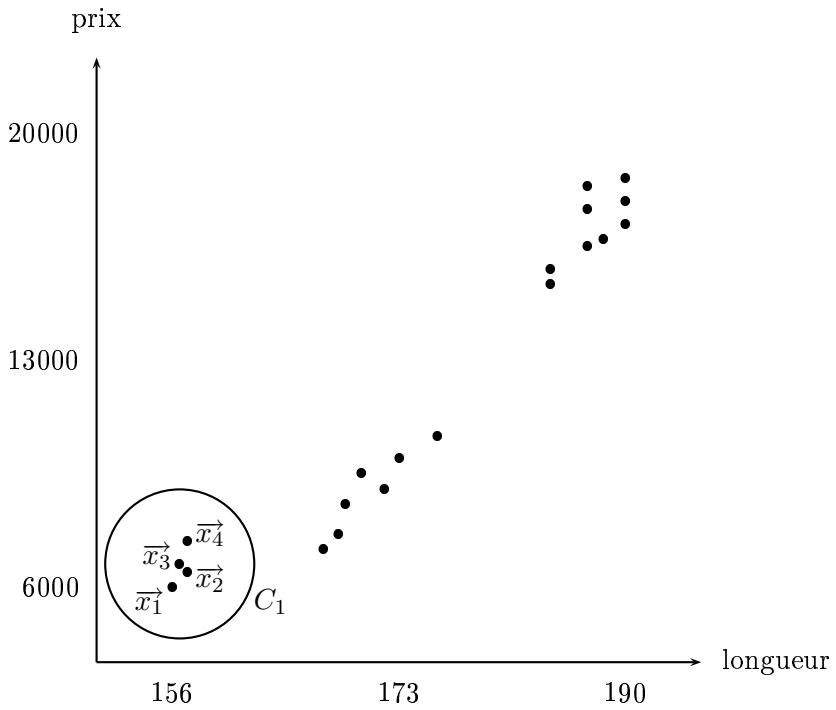


FIG. 2.5 – Exemple de notations.

Il existe différentes façons de représenter les clusters C_k, les représentations les plus utilisées classiquement étant les suivantes :

1. son *centroïde* $\overrightarrow{\mu_k}$, centre de gravité du cluster :

$$\overrightarrow{\mu_k} = \frac{1}{N_k} \sum_{i \in D_k} \overrightarrow{x_i}$$

2. l'un des membres du cluster, proche de son centre de gravité, appelé *médoïde* ;
3. un ensemble d'objets distants appartenant au cluster : typiquement ceux qui se trouvent aux frontières du cluster :

$$\{i \in D_k | \ \exists\ d \in S\ \forall\ j \in D_k, x_{id} \geq x_{jd}\} \cup \{i \in D_k | \ \exists\ d \in S\ \forall\ j \in D_k, x_{id} \leq x_{jd}\}$$

4. une expression logique conjonctive : typiquement le *moindre généralisé* des objets appartenant au cluster, hyperrectangle minimal contenant l'ensemble des membres du cluster :

$$\{[min_d, max_d] \mid d \in S \text{ et } min_d = min_{\{i \in D_k\}} x_{id} \text{ et } max_d = max_{\{i \in D_k\}} x_{id}\}$$

5. ou statistiquement : typiquement son centroïde et sa matrice de covariance.

Différentes mesures de distance peuvent être utilisées pour définir la similarité entre objets. La mesure la plus utilisée est indubitablement la distance euclidienne, présentée ci-après. Mais d'autres mesures peuvent également être envisagées. Nous en présentons davantage en annexes A.1.1.

$$dist(\overrightarrow{x_i}, \overrightarrow{x_j}) = \sqrt{\sum_{d=1}^{M} (x_{id} - x_{jd})^2}$$

De même, si différentes mesures peuvent être utilisées pour caractériser la *cohésion interne* d'un cluster, la mesure la plus souvent utilisée est le radius du cluster, distance moyenne entre membres du cluster et son centroïde, à minimiser pour maximiser la similarité des observations à l'intérieur du cluster :

$$Radius(C_k) = \frac{1}{N_k} \sum_{i \in D_k} dist(\overrightarrow{x_i}, \overrightarrow{\mu_k})$$

Basée sur le même principe, une mesure utilisée classiquement pour évaluer la *qualité interne* d'un clustering est l'erreur quadratique, les solutions qui minimisent cette mesure étant alors considérées comme les plus pertinentes :

$$E^2(P) = \sum_{k=1}^{K} \sum_{i \in D_k} dist(\overrightarrow{x_i}, \overrightarrow{\mu_k})$$

Enfin, si l'on souhaite évaluer la qualité d'un clustering en fonction des classes connues de chaque donnée, deux mesures de la *qualité externe* du clustering sont classiquement utilisées : la *F-mesure* et l'*Entropie*. Ces mesures sont basées sur deux notions

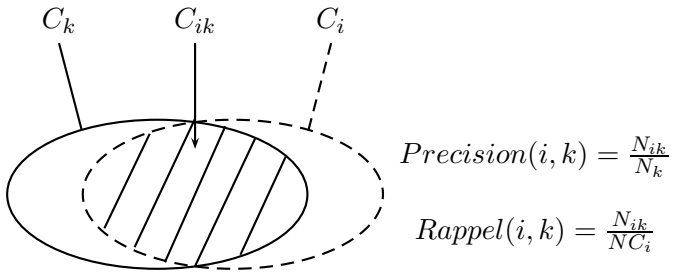


FIG. 2.6 – Illustration des notions de Rappel et de Précision.

qui sont illustrées par la figure 2.6 : le *Rappel* et la *Précision*, NC_i dénotant le nombre de membres de la classe i, N_k le nombre de membres du cluster C_k, et N_{ik} le nombre de membres de la classe i dans le cluster C_k.

La F-mesure est calculée comme suit :

$$F(P) = \sum_i \frac{NC_i}{N} Max_{k=1}^{K} F(i,k)$$

$$F(i,k) = \frac{(1+\beta) \times Rappel(i,k) \times Precision(i,k)}{\beta \times Rappel(i,k) + Precision(i,k)}$$

où β est utilisé pour contrôler l'importance du Rappel sur la Précision, typiquement positionné à 1 de façon à ce que le Rappel et la Précision aient la même importance :

$$F(i,k) = \frac{2 \times Rappel(i,k) \times Precision(i,k)}{Rappel(i,k) + Precision(i,k)}$$

L'Entropie E_k du cluster C_k est quant à elle calculée comme suit :

$$E_k = -\sum_i Precision(i,k) \times \log Precision(i,k)$$

Et finalement, l'Entropie d'une partition est évaluée par la somme des Entropies de chaque cluster, pondérée par leur taille :

$$E(P) = \sum_{k=1}^{K} \frac{N_k \times E_k}{N}$$

La partition P qui maximise la F-mesure associée, ou minimise l'Entropie associée, est alors considérée comme la plus pertinente car c'est celle qui correspond le mieux à la solution externe attendue.

2.7 Bilan

Dans ce chapitre, nous avons donc vu qu'il existe différentes applications au clustering, chacune ayant un objectif différent, et donc un critère de la qualité du clustering qui peut être différent. La première étape de la mise en œuvre ou de l'utilisation d'une méthode de clustering doit par conséquent être d'identifier l'application visée afin d'identifier ses besoins.

Nous avons également vu qu'il existe différentes notions possibles de la similarité entre objets, davantage de détails sur ce point étant disponibles en annexes A.1. Là encore, le choix important de la fonction de distance utilisée pour évaluer la similarité entre objets doit être effectué au préalable en fonction du problème considéré.

Il existe aussi de nombreuses méthodes de clustering différentes avec chacune ses atouts et ses limites. Le chapitre suivant est consacré à la présentation plus détaillée des méthodes existantes les plus classiques en clustering, davantage de détails étant également disponibles en annexes A.

Dans la partie II qui suit, nous proposons ensuite deux nouvelles méthodes de clustering, qui prennent en compte la problématique récente et importante du contexte que nous avons évoquée, et qui présentent certains atouts par rapport aux méthodes existantes.

L'application que nous visons concerne l'extraction de connaissances. Nous portons donc une attention particulière à la minimisation du nombre de connaissances a priori requises de la part de l'utilisateur, et à la présentation des résultats sous un format facilement interprétable par l'utilisateur.

Enfin, la partie III suivante aborde la problématique ouverte importante de l'évaluation dans le cadre du clustering. Là encore, nous verrons que la nouvelle méthode que nous proposons possède certains avantages par rapport aux méthodes existantes.

Chapitre 3

Méthodes de clustering

Face au nombre important de méthodes de clustering qui ont été développées et d'alternatives qui ont été proposées, en fournir une liste exhaustive demanderait sûrement plus de trois ans d'investigations et serait probablement très difficile à intégrer pour le lecteur. Dans ce chapitre, nous avons donc choisi de présenter un tour d'horizon des méthodes les plus classiques. Plus de détails sont ensuite proposés en annexes A.

Pour chacune des méthodes, nous présentons d'abord son fonctionnement général, avec éventuellement différentes alternatives possibles, puis nous montrons son application sur l'exemple concernant les voitures présenté dans la figure 1.1 du chapitre 1. Nous détaillons ensuite ses atouts et ses limites, que nous résumons finalement à l'aide du tableau des caractéristiques associées aux méthodes de clustering que nous avons proposé dans le chapitre précédent.

3.1 Clustering hiérarchique

Le fondement du clustering hiérarchique est de créer une hiérarchie de clusters. À la racine de l'arbre est associé un unique cluster contenant l'ensemble des objets de la base, puis plus on descend dans l'arbre, plus les clusters sont spécifiques à un certain groupe d'objets considérés comme similaires. La figure 3.1 montre une telle hiérarchie dans le cas de notre exemple concernant les voitures.

Afin de former une telle hiérarchie de clusters, il existe deux méthodes principales :

1. la méthode ascendante, démarrant avec autant de clusters que d'objets initiaux dans la base, puis fusionnant successivement les clusters considérés comme les plus similaires, jusqu'à ce que tous les objets soient réunis dans un unique cluster stocké à la racine de la hiérarchie formée ;
2. et la méthode descendante, démarrant avec un unique cluster contenant l'ensemble des objets de la base, puis divisant successivement les clusters de manière à ce que les clusters résultants soient les plus différents possible, et ce jusqu'à obtenir aux feuilles de la hiérarchie autant de clusters que d'objets dans la base.

On retrouve donc à ce niveau la différence classique en apprentissage entre généralisation et spécialisation : les méthodes ascendantes démarrent avec une solution tout

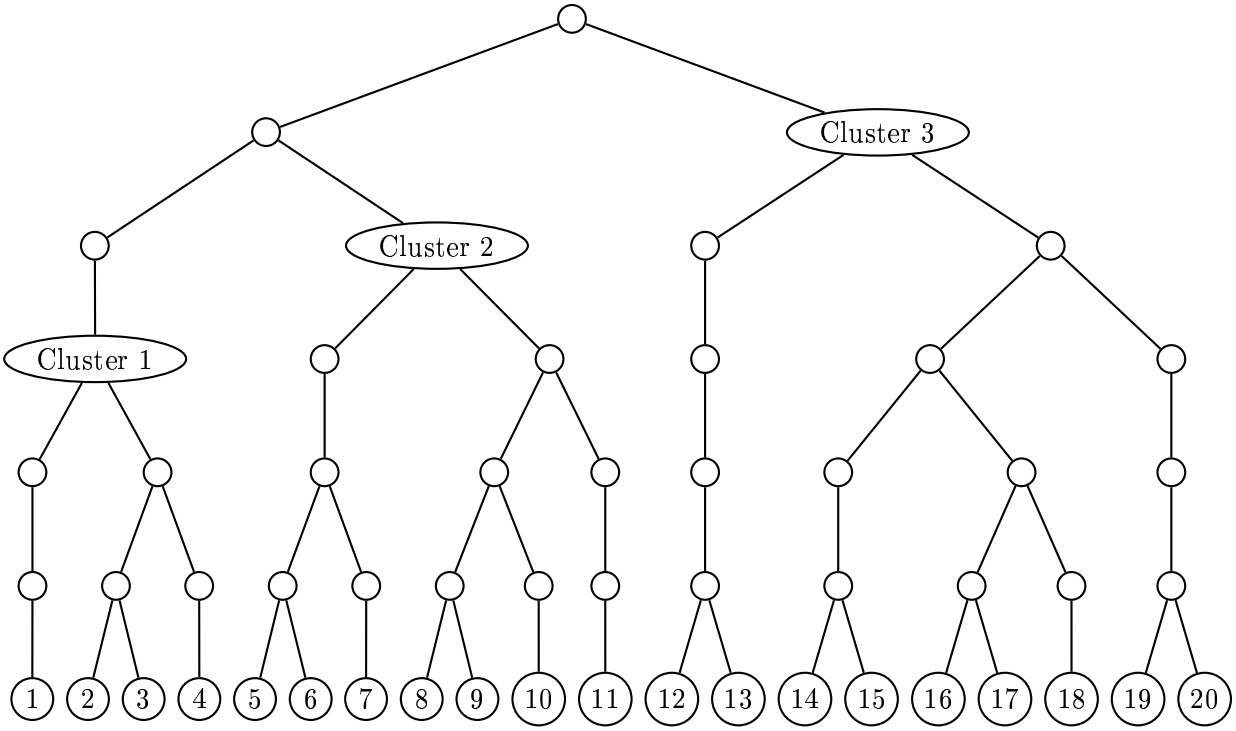


FIG. 3.1 – Clustering hiérarchique sur l'exemple.

à fait spécifique aux données, qui est ensuite généralisée à chaque étape, alors que les méthodes descendantes démarrent avec une solution complètement générale, qui est ensuite spécialisée à chaque étape.

Étant donnée une mesure de distance entre deux clusters, la fusion suivante à effectuer pour une méthode ascendante concernera les deux clusters les plus proches, alors que la division suivante à effectuer pour une méthode descendante concernera les deux clusters les plus éloignés. Trois alternatives principales existent pour définir la distance entre deux clusters :

1. pour la méthode *single-link*, la distance entre deux clusters est le minimum des distances entre toutes les paires d'objets appartenant à des clusters différents ;
2. pour la méthode *complete-link*, on utilise le maximum de ces distances ;
3. et pour la méthode *average-link*, la moyenne de ces distances.

Ensuite, une fois la hiérarchie formée, une étape optionnelle peut être ajoutée pour affiner le résultat fourni à l'utilisateur. Il s'agit alors de déterminer le niveau de coupure le plus approprié à appliquer dans l'arbre pour un regroupement des données aussi pertinent que possible. Pour cela, on peut demander à l'utilisateur de spécifier le nombre de clusters attendus, et utiliser les différentes mesures de la qualité interne des différentes partitions possibles pour sélectionner la plus pertinente. Ou bien on peut demander à l'utilisateur de spécifier des seuils sur la cohésion interne ou l'isolation externe minimales des clusters.

Bien que largement utilisée, cette méthode devient difficilement utilisable face à de larges bases de données, car sa complexité est quadratique en fonction du nombre d'objets de la base, puisque les distances entre toutes les paires d'objets possibles doivent être calculées. Par ailleurs, cette méthode ne peut jamais défaire ce qu'elle a déjà fait auparavant. Elle ne peut donc être utilisée de façon incrémentale. Enfin, elle souffre du problème de l'*effet de chaîne*, c'est-à-dire que des clusters proches mais distincts peuvent être fusionnés s'il existe une chaîne d'objets qui les relie. Finalement, le tableau 3.1 résume les différentes caractéristiques de cette approche.

3.2 Clustering K-means

Étant donné le nombre K de clusters recherchés, la méthode de clustering K-means [Diday et al., 1982b], illustrée par la figure 3.2, est la suivante :

1. choisir aléatoirement K objets de la base qui formeront l'ensemble des centroïdes initiaux représentant les K clusters recherchés ;
2. assigner chaque objet au cluster dont le centroïde est le plus proche ;
3. puis tant qu'au moins un objet change de cluster d'une itération à l'autre :
   - mettre à jour les centroïdes des clusters en fonction des objets qui leur sont associés :

$$\overrightarrow{\mu_k} = \frac{1}{N_k} \sum_{i \in D_k} \overrightarrow{x_i}$$

caractéristique	**valeur**
connaissances a priori	nombre de clusters ou seuils
présentation des résultats	hiérarchie
complexité	$O(M \times N^2)$
déterministe	oui
incrémental	non
any-time	non
hard	oui
prise en compte du contexte	non
tolérance au bruit	non
tolérance à l'effet de chaîne	non
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	oui
tolérance aux clusters de forme quelconque	oui
tolérance aux clusters concentriques	oui

TAB. 3.1 – Caractéristiques associées au clustering hiérarchique.

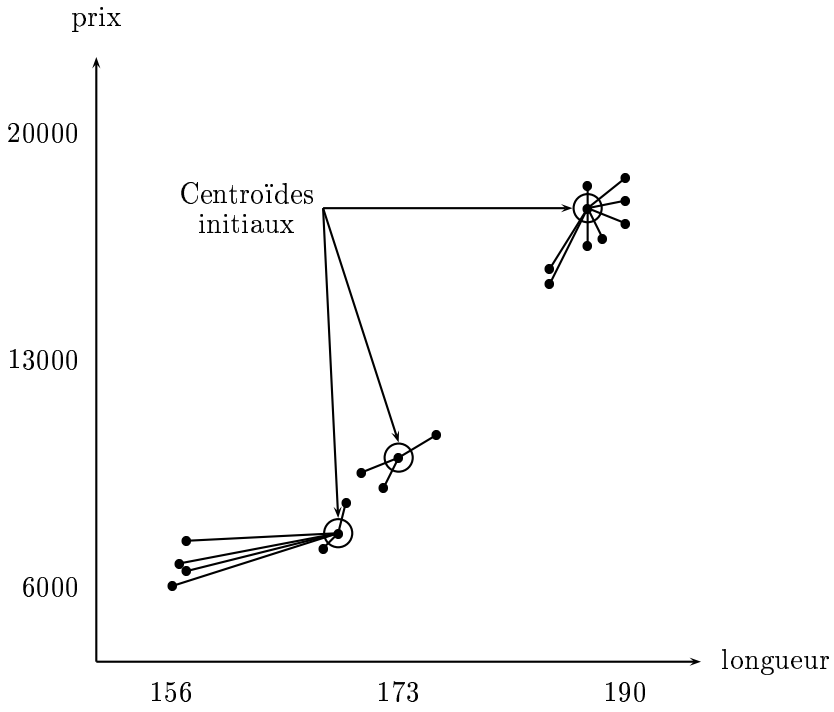
– mettre à jour les assignations des objets aux clusters en fonction de leur proximité aux nouveaux centroïdes :

$$D_k = \{i \in D | \ \forall \ j \in [1, K] \text{ et } j \neq k, dist(\overrightarrow{x_i}, \overrightarrow{\mu_k}) < dist(\overrightarrow{x_i}, \overrightarrow{\mu_j})\}$$

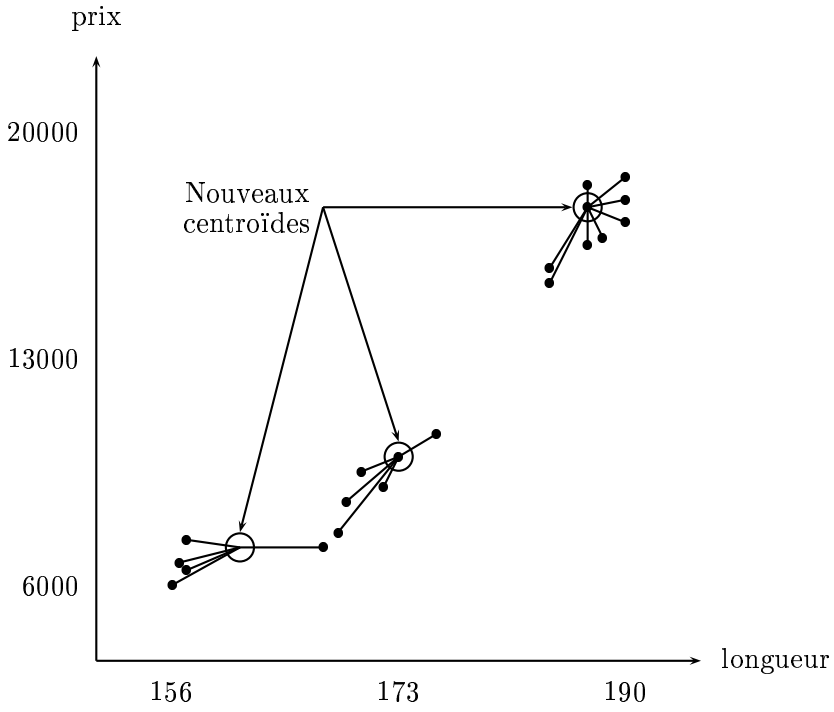
La troisème étape de cette procédure peut être effectuée de manière incrémentale ou non : soit chaque fois qu'un changement d'assignation d'un objet est rencontré, les deux centroïdes des clusters concernés par le changement sont mis à jour, soit tous les centroïdes sont mis à jour simultanément après toutes les assignations des objets aux clusters.

Une problématique importante dans ce cadre concerne la spécification du nombre de clusters recherchés. Malheureusement, dans de nombreux cas, cette information n'est pas connue. Typiquement, la solution choisie pour résoudre ce problème est de générer plusieurs partitions en modifiant le nombre de clusters recherchés, puis d'utiliser un test statistique permettant de sélectionner le partitionnement le plus approprié aux données [Hamerly and Elkan, 2003]. Dans la construction de ce test statistique, on retrouve le compromis classique entre spécificité et généralisation du modèle choisi, puisqu'un tel critère est classiquement composé d'un terme reflétant la qualité interne du clustering et d'un autre terme qui pénalise les modèles plus complexes (ceux basés sur un nombre plus important de clusters, qui sont donc plus spécifiques).

Cette méthode est aussi très sensible à la sélection initiale des centroïdes et converge vers des solutions optimales locales. Dans la plupart des cas, la solution choisie pour pallier à cette limitation est d'itérer la méthode en faisant varier de façon aléatoire la sélection initiale des centroïdes, puis de sélectionner le résultat optimisant un certain critère de qualité interne.



(a) Étapes 1 et 2.



(b) Une itération de l'étape 3.

FIG. 3.2 – Clustering K-means sur l'exemple.

De plus, cette méthode ne produit ainsi que des clusters de forme hypersphérique et n'est pas capable de gérer des clusters proches dont les tailles sont très différentes. Une solution pour pallier à ce problème est de permettre de diviser ou de fusionner a posteriori les clusters obtenus. Typiquement, un cluster est divisé quand sa variance est au-dessus d'un certain seuil préspécifié, et deux clusters sont fusionnés lorsque la distance entre leurs centroïdes est en-dessous d'un autre seuil préspécifié. Une autre solution consiste à utiliser un modèle plus riche pour la représentation des clusters. Nous verrons dans la section suivante que le clustering statistique généralise le clustering K-means [Celeux and Govaert, 1992]. Les *nuées dynamiques* constituent également une méthode qui généralise la méthode K-means [Diday et al., 1980].

Par contre, cette méthode a l'avantage important d'avoir une complexité linéaire par rapport au nombre d'objets fournis en entrée de l'algorithme. Finalement, le tableau 3.2 résume les différentes caractéristiques de cette approche.

caractéristique	**valeur**
connaissances a priori	nombre de clusters K
présentation des résultats	K centroïdes
complexité	$O(M \times N \times K)$
déterministe	non
incrémental	oui
any-time	oui
hard	oui
prise en compte du contexte	non
tolérance au bruit	non
tolérance à l'effet de chaîne	oui
tolérance aux clusters de tailles variées	non
tolérance aux clusters de densités variées	oui
tolérance aux clusters de forme quelconque	non
tolérance aux clusters concentriques	non

TAB. 3.2 – Caractéristiques associées au clustering K-means.

3.3 Clustering statistique

L'approche statistique du clustering consiste à supposer que les données ont été générées selon une loi paramétrique connue, mais de paramètres inconnus. On considère que les données ont été générées par un mélange de distributions de probabilités, la problématique étant alors de trouver, en fonction des données observées, les paramètres de ces distributions, ainsi que les probabilités du mélange, et les paramètres cachés du modèle correspondant aux affectations des objets aux différentes composantes du mélange.

Par exemple, lorsque les données sont de type numérique, on suppose classiquement

qu'elles ont été générées selon un mélange de distributions gaussiennes [Ye and Spetsakis, 2003], caractérisées par leur centre et leur matrice de covariance, chacune de ces distributions ayant une certaine probabilité d'être utilisée.

On construit donc un modèle des données θ qui représente un mélange de K distributions gaussiennes θ_k pour $k \in [1, K]$, caractérisées par leur centre $\overrightarrow{\mu_k}$, leur matrice de covariance Σ_k, et leur probabilité d'apparition π_k.

$$\theta = (\theta_1, ..., \theta_K)$$

$$\theta_k = (\overrightarrow{\mu_k}, \Sigma_k, \pi_k)$$

$$\sum_{k=1}^{K} \pi_k = 1$$

Les appartenances des objets aux clusters sont alors vues comme des paramètres cachés du modèle. On pose $z_{ik} = 1$ si l'objet $\overrightarrow{x_i}$ a été généré par la distribution θ_k, et $z_{ik} = 0$ sinon. Puis l'objectif est de maximiser la vraisemblance du modèle θ par rapport aux données D, notée $L(\theta|D)$. On parle donc d'*estimation par maximum de vraisemblance*. Si l'on fait l'hypothèse que les objets ont été générés indépendamment les uns des autres, on a :

$$L(\theta|D) = P(D|\theta) = \prod_{i=1}^{N} P(\overrightarrow{x_i}|\theta)$$

L'utilisation du logarithme de cette fonction permet de considérer une somme de termes au lieu d'un produit, et maximiser $L(\theta|D)$ équivaut à maximiser son logarithme.

$$\log L(\theta|D) = \sum_{i=1}^{N} \log P(\overrightarrow{x_i}|\theta)$$

$$P(\overrightarrow{x_i}|\theta) = \sum_{k=1}^{K} \pi_k \times P(\overrightarrow{x_i}|\theta_k)$$

Avec, dans le cas gaussien :

$$P(\overrightarrow{x_i}|\theta_k) = \frac{1}{\sqrt{(2\pi)^M |\Sigma_k|}} e^{-\frac{1}{2}(\overrightarrow{x_i} - \overrightarrow{\mu_k})^T \Sigma_k^{-1} (\overrightarrow{x_i} - \overrightarrow{\mu_k})}$$

Une méthode classique pour résoudre ce problème d'optimisation est la méthode *EM* [Dempster et al., 1977]. Celle-ci consiste à itérer deux phases permettant d'améliorer la vraisemblance du modèle par rapport aux données à chaque étape :

1. *Espérance* : supposer fixés les paramètres du modèle θ, et chercher l'ensemble des affectations des objets aux clusters optimales sous ce modèle. On pose alors $\overline{z}_{ik}$ l'espérance de la valeur de z_{ik} :

$$\overline{z}_{ik} = \frac{\pi_k \times P(\overrightarrow{x_i}|\theta_k)}{\sum_j \pi_j \times P(\overrightarrow{x_i}|\theta_j)}$$

   Les valeurs des $\overline{z}_{ik}$ sont donc définies dans l'intervalle $[0, 1]$, alors que celles des z_{ik} sont définies dans $\{0, 1\}$, mais pour un objet donné $\overrightarrow{x_i}$, on conserve la propriété suivante :

$$\sum_{k=1}^{K} \overline{z}_{ik} = 1$$

2. *Maximisation* : supposer fixées les affectations des objets aux clusters, et calculer les paramètres optimaux du modèle θ en fonction de cette connaissance.

$$\pi_k = \frac{\sum_{i=1}^{N} \overline{z}_{ik}}{N}$$

   Dans le cas d'un mélange de K gaussiennes, chaque composante du modèle dépend d'un centre $\overrightarrow{\mu_k}$ et d'une matrice de covariance Σ_k qui sont mis à jour comme suit :

$$\overrightarrow{\mu_k} = \frac{\sum_{i=1}^{N} \overline{z}_{ik} \times \overrightarrow{x_i}}{N \times \pi_k}$$

$$\Sigma_k = \frac{1}{N \times \pi_k} \sum_{i=1}^{N} \overline{z}_{ik} (\overrightarrow{x_i} - \overrightarrow{\mu_k})(\overrightarrow{x_i} - \overrightarrow{\mu_k})^T$$

Cette procédure s'arrête lorsque d'une itération à l'autre, la différence entre deux valeurs successives de la log vraisemblance $\log L(\theta|D)$ est inférieure à un seuil fixé δ strictement positif.

En fait, ce modèle généralise celui utilisé par la méthode K-means. En effet, si l'on fait l'hypothèse que les matrices de covariance des clusters Σ_k correspondent toutes à la matrice identité, et que les probabilités du mélange π_k sont toutes égales, alors on obtient le modèle utilisé par la méthode K-means [Celeux and Govaert, 1992].

Ainsi, de même que pour la méthode K-means, cette méthode statistique est confrontée à la problématique de la spécification du nombre de clusters recherchés, et est également sensible à la solution initiale proposée. Les solutions pour pallier à ces limitations sont donc les mêmes : générer plusieurs modèles en faisant varier le nombre de clusters recherchés, et utiliser un critère statistique pour choisir le modèle le plus approprié aux données, en considérant pour chaque modèle plusieurs initialisations aléatoires.

Plusieurs critères ont été proposés pour cela qui ajoutent au calcul de la vraisemblance du modèle par rapport aux données un terme qui tend à pénaliser les modèles les plus complexes. Les différents critères se différencient donc sur leur façon de pénaliser

les modèles les plus complexes. Parmi ceux-là, on peut par exemple citer le critère AIC (Akaike Information Criterion) [Akaike, 1970], ou bien le critère ICL (Integrated Completed Likelihood) [Biernacki et al., 2000], mais le critère le plus connu et le plus utilisé dans ce cadre est sans doute le critère BIC (Bayesian Information Criterion) [Schwartz, 1979], défini comme suit :

$$BIC(\theta|D) = -2 \times \log L(\theta|D) + M_\theta \times \log N$$

Le premier terme de ce critère évalue la vraisemblance du modèle par rapport aux données, c'est-à-dire la spécificité du modèle, tandis que le second terme pénalise les modèles plus complexes, c'est-à-dire trop spécifiques, M_θ représentant le nombre de paramètres indépendants du modèle. La valeur de ce critère doit être minimisée pour optimiser la pertinence du modèle par rapport aux données.

Par contre, contrairement à K-means, cette méthode statistique est capable d'identifier des clusters de forme allongée, et est capable de gérer des clusters proches dont les tailles sont très différentes. Elle est également naturellement capable de gérer le bruit qui peut exister dans les données. Cependant, elle nécessite souvent de nombreuses itérations avant de converger et sa complexité est quadratique en fonction du nombre de dimensions M. Finalement, le tableau 3.3 résume les différentes caractéristiques de cette approche.

caractéristique	**valeur**
connaissances a priori	nombre de clusters K
présentation des résultats	K centroïdes et matrices de covariances
complexité	$O(M^2 \times N \times K)$
déterministe	non
incrémental	oui
any-time	oui
hard	non
prise en compte du contexte	non
tolérance au bruit	oui
tolérance à l'effet de chaîne	oui
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	oui
tolérance aux clusters de forme quelconque	non
tolérance aux clusters concentriques	non

TAB. 3.3 – Caractéristiques associées au clustering statistique.

3.4 Clustering stochastique

Étant donnée une mesure de la qualité interne d'un clustering considérée, par exemple l'erreur quadratique présentée dans la section 2.6 du chapitre précédent, plusieurs mé-

thodes de recherche stochastique de la partition des objets optimisant ce critère peuvent être envisagées [Berkhin, 2002].

Dans ce cas, la technique consiste simplement à parcourir l'espace des solutions possibles et à sélectionner la solution rencontrée qui maximise le critère cible choisi. Évidemment, l'espace des solutions étant souvent beaucoup trop vaste pour être parcouru entièrement, des heuristiques sont généralement utilisées pour le parcourir.

Typiquement, une solution considérée dans ce cadre est l'ensemble des associations des objets aux clusters, le nombre de clusters étant fourni par l'utilisateur. À chaque étape, une nouvelle solution est considérée, et conservée pour l'étape suivante si elle fait partie des solutions optimales du point de vue du critère cible.

L'originalité des différentes méthodes stochastiques existantes réside alors dans leur façon de parcourir l'espace des solutions possibles, c'est-à-dire dans la technique utilisée pour proposer une nouvelle solution courante à évaluer. En pratique, les méthodes mises en œuvre sont souvent construites de manière à trouver un bon compromis entre l'exploration nécessaire d'une partie importante de l'espace des solutions, et l'exploitation des solutions optimales identifiées.

Ainsi, la technique de la *recherche Tabou* sélectionne pour l'étape suivante une solution voisine de la solution optimale courante, en s'empêchant de considérer une solution déjà rencontrée précédemment.

La technique de *recuit simulé* commence par considérer des solutions très différentes les unes des autres, afin de parcourir le plus largement possible l'espace des solutions, puis restreint au fur et à mesure sa recherche aux solutions voisines de la solution optimale courante.

Enfin, la technique des *algorithmes génétiques* considère plutôt des populations de solutions potentielles, et combine ces différentes solutions pour obtenir à l'étape suivante un autre ensemble potentiel de solutions, parmi lesquelles sont conservées celles qui optimisent le critère cible choisi.

Une autre technique envisagée dans de nombreux cas est de considérer qu'une solution correspond à un ensemble de K médoïdes, c'est-à-dire un ensemble de K objets de la base servant de points centraux des clusters à identifier. Les autres objets sont alors associés au cluster dont le médoïde est le plus proche. Et une méthode pour parcourir l'espace des solutions possibles consiste alors à modifier l'un des médoïdes, et à le conserver pour l'itération suivante si la solution est améliorée par ce changement.

Les caractéristiques de telles méthodes dépendent très fortement du critère cible choisi. Ainsi, si c'est l'erreur quadratique qui est retenue, alors la méthode ne peut faire face à des clusters proches dont les tailles sont très variées. Si au contraire le critère choisi est un test statistique basé sur une distribution gaussienne des données, associant à chaque cluster un centre et une matrice de covariance, alors la méthode sera capable de gérer de tels clusters. C'est en considérant ce dernier cas que le tableau 3.4 résume les différentes caractéristiques de la méthode.

caractéristique	**valeur**
connaissances a priori	nombre de clusters K
présentation des résultats	partition
complexité	$O(M \times N \times K)$
déterministe	non
incrémental	oui
any-time	oui
hard	oui
prise en compte du contexte	non
tolérance au bruit	oui
tolérance à l'effet de chaîne	oui
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	oui
tolérance aux clusters de forme quelconque	non
tolérance aux clusters concentriques	non

TAB. 3.4 – Caractéristiques associées au clustering stochastique.

3.5 Clustering basé sur la densité

Une autre vision possible des clusters est de les considérer comme des régions homogènes de haute densité, entourées par des régions de faible densité. Basée sur cette vision des clusters, une autre famille de méthodes est apparue, fondée sur la formation des clusters de manière *gloutonne* à partir d'un certain critère de densité.

La méthode de référence dans ce cadre, appelée DBSCAN [Ester et al., 1996], est ainsi basée sur l'utilisation de deux paramètres fournis en entrée de l'algorithme et contrôlant la notion de densité du voisinage d'un objet :

1. *Eps* : le radius du voisinage de l'objet ;
2. et *MinPts* : le nombre minimum d'objets qui doivent être contenus dans ce voisinage pour considérer la zone comme dense.

Illustrée par la figure 3.3, la méthode consiste en une succession de deux phases itérées jusqu'à ce que tous les objets soient assignés à un cluster :

1. sélectionner l'un des objets $\overrightarrow{x_i}$ de la base qui n'est encore assigné à aucun cluster ;
2. si son voisinage respecte le critère de densité, c'est-à-dire si au moins *MinPts* objets sont présents dans l'hypersphère de centre $\overrightarrow{x_i}$ et de rayon *Eps*, alors les objets correspondants sont intégrés au cluster courant et cette même phase est itérée pour ces objets.

Un avantage important de cette méthode est qu'elle permet de considérer des clusters de formes très variées ainsi que des clusters concentriques (c'est-à-dire contenus les uns dans les autres). Elle est également naturellement capable de faire face au bruit qui peut exister dans les données.

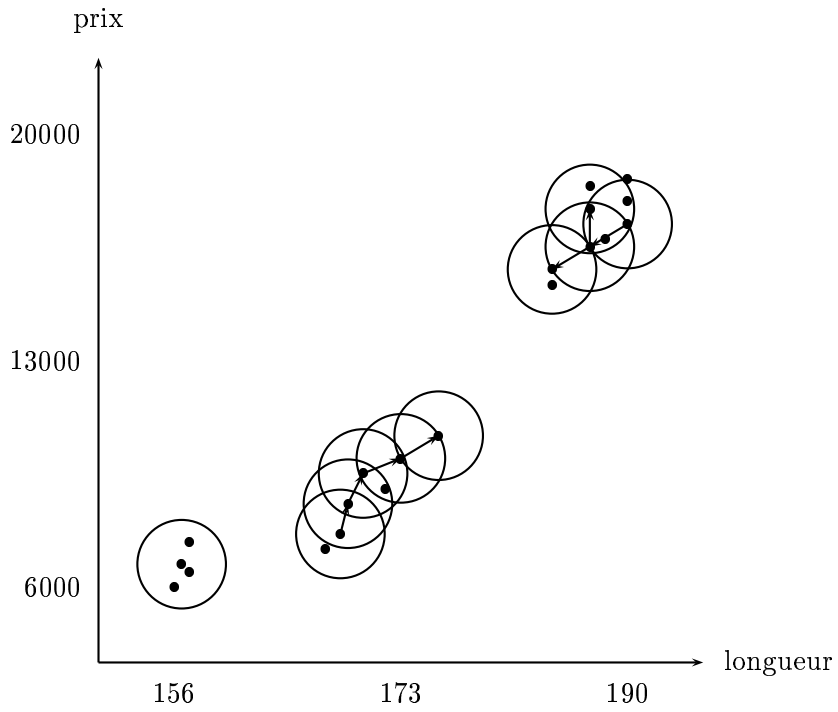


FIG. 3.3 – Clustering basé sur la densité sur l'exemple.

Par contre, sa complexité est quadratique en fonction du nombre d'objets de la base. De plus, elle souffre du problème d'*effet de chaîne*. Par ailleurs, elle est sensible aux valeurs fixées pour les deux paramètres *Eps* et *MinPts*. Or il est souvent très difficile à un utilisateur de spécifier les valeurs les plus appropriées pour ces paramètres. De plus, ces paramètres étant fixés de manière globale, ils ne permettent pas de cibler des clusters dont les densités sont très différentes. Le tableau 3.5 résume finalement les caractéristiques d'une telle méthode.

caractéristique	**valeur**
connaissances a priori	critère de densité du voisinage
présentation des résultats	partition
complexité	$O(M \times N^2)$
déterministe	oui
incrémental	non
any-time	non
hard	oui
prise en compte du contexte	non
tolérance au bruit	oui
tolérance à l'effet de chaîne	non
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	non
tolérance aux clusters de forme quelconque	oui
tolérance aux clusters concentriques	oui

TAB. 3.5 – Caractéristiques associées au clustering basé sur la densité.

3.6 Clustering basé sur les grilles

Une autre approche possible pour effectuer le clustering d'un ensemble de données est d'utiliser une grille pour partitionner l'espace en de nombreuses cellules, puis d'identifier les ensembles de cellules denses connectées pour former les clusters [Brézellec and Didier, 2001].

Dans ce cas, l'utilisateur doit spécifier la taille de la grille à utiliser et la densité minimum déterminant si une cellule de la grille est considérée comme dense ou non. Dans un premier temps, l'espace de description des objets est découpé par des intervalles réguliers sur chaque dimension. Puis la densité de chaque cellule est évaluée, et les cellules satisfaisant le critère de densité minimum sont détectées. Ensuite, les composantes connexes, c'est-à-dire les ensembles de cellules denses connectées, sont identifiées. Enfin, un test d'uniformité de ces composantes connexes est mené. Les composantes connexes uniformes forment alors les clusters, et pour les autres, la cellule la moins dense est supprimée, et le test est reconduit sur chacune des deux sous-composantes obtenues jusqu'à ce que toutes les composantes soient considérées comme uniformes.

La figure 3.4 présente cette méthode sur l'exemple. On peut dès lors observer qu'elle admet que certains objets soient isolés. C'est le cas de l'objet caractérisé par une longueur de 176 et un prix de 10600. Dans ce cas, ceci est la conséquence du choix de la taille de grille utilisée, mais dans d'autres cas, cela permet d'isoler le bruit qui peut exister dans les données.

Pour ces méthodes, la problématique la plus critique est celle du choix du critère de densité et de la taille de la grille utilisée :

- des cellules de trop petite taille amènent à une estimation bruitée de la densité : c'est le problème du *sur-partitionnement* ;
- à l'inverse, des cellules de trop grande taille amènent à de trop faibles densités : c'est le problème du *sous-partitionnement*.

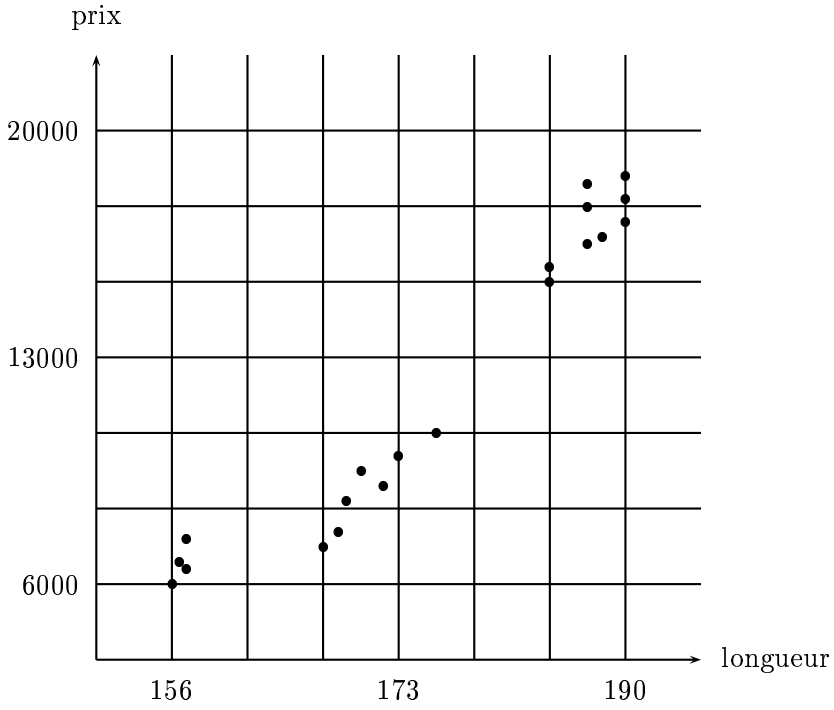
Par ailleurs, ces techniques ont les mêmes forces et faiblesses que les méthodes basées sur la densité : elles souffrent de l'*effet de chaîne* et prennent difficilement en compte le fait que des clusters de densités différentes puissent exister, mais elles sont capables d'identifier des clusters de formes très variées ou concentriques et de gérer le bruit qui peut exister dans les données. Le tableau 3.6 résume les caractéristiques de cette approche.

caractéristique	**valeur**
connaissances a priori	taille de grille et critère de densité
présentation des résultats	ensembles de cellules connectées
complexité	$O(M \times \textit{taille de grille})$
déterministe	oui
incrémental	non
any-time	non
hard	oui
prise en compte du contexte	non
tolérance au bruit	oui
tolérance à l'effet de chaîne	non
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	non
tolérance aux clusters de forme quelconque	oui
tolérance aux clusters concentriques	oui

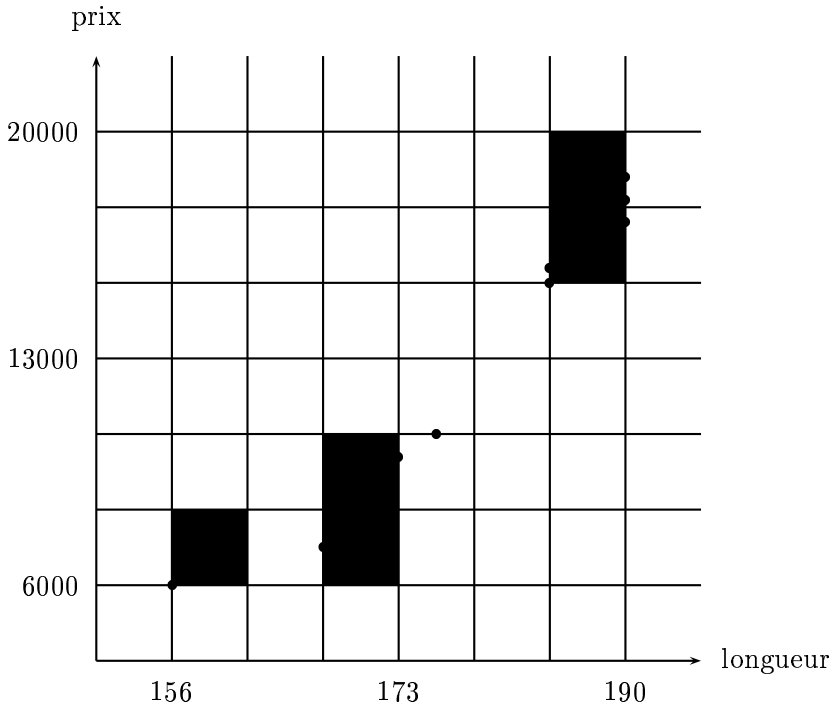
TAB. 3.6 – Caractéristiques associées au clustering basé sur les grilles.

3.7 Clustering basé sur les graphes

Illustrée par la figure 3.5, la méthode de clustering basée sur les graphes consiste à considérer les clusters comme des ensembles de nœuds connectés dans un graphe. Typiquement, on construit dans un premier temps le graphe complet des données, dans lequel un nœud correspond à un objet, et un arc à la distance entre les deux objets



(a) Partitionnement de l'espace par l'utilisation d'une grille.



(b) Et le clustering associé.

FIG. 3.4 – Clustering basé sur les grilles sur l'exemple.

considérés. On en dérive ensuite le *Minimal Spanning Tree* (MST) : graphe connexe dont la somme des valeurs des arcs, correspondant aux distances dans notre cas, est minimale. Puis selon le paramètre utilisé pour la formation des clusters, il peut s'agir soit de supprimer les arcs les plus longs, soit de conserver les arcs dont la valeur est inférieure à un seuil spécifié.

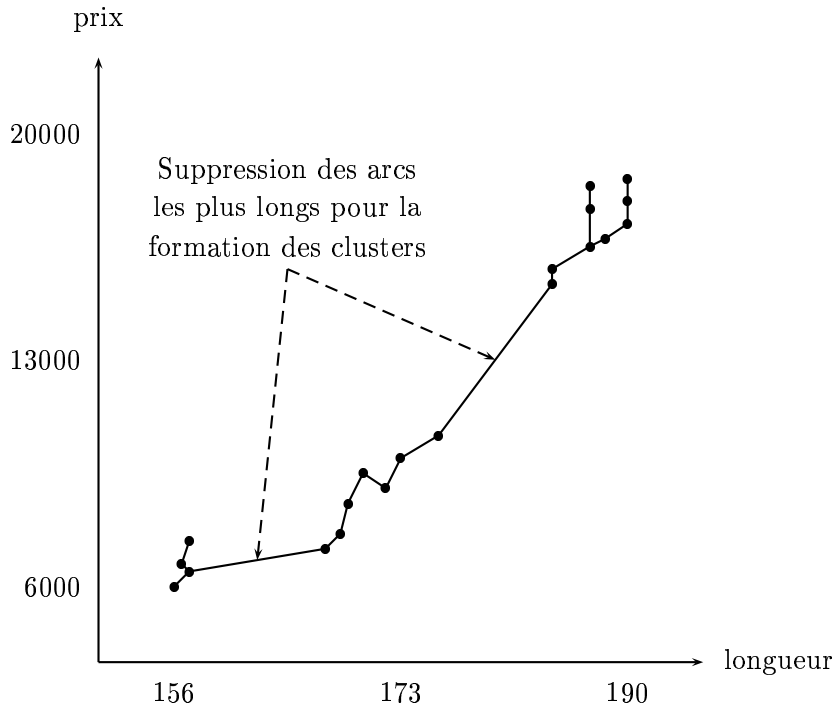


FIG. 3.5 – Clustering basé sur les graphes sur l'exemple.

La limite ici est principalement la complexité de la méthode utilisée pour former le graphe MST initial : en utilisant une méthode classique de construction de MST, la complexité est en $O(N^2 \log N)$. Différentes techniques utilisées pour réduire cette complexité peuvent être trouvées dans [Pettie and Ramachandran, 2000], mais celle-ci sera de toute façon au moins quadratique dans le meilleur des cas. Finalement, le tableau 3.7 résume les caractéristiques de cette méthode.

3.8 Clustering spectral

Partant d'une matrice de similarité entre les objets, le fondement du clustering spectral consiste à projeter les objets dans un sous-espace de variance maximum, puis d'utiliser une méthode de partitionnement dans un tel sous-espace pour identifier les clusters [Verma and Meila, 2003].

caractéristique	**valeur**
connaissances a priori	distance minimale entre clusters
présentation des résultats	partition sous forme de graphe
complexité	$O(M \times N^2 \log N)$
déterministe	oui
incrémental	non
any-time	non
hard	oui
prise en compte du contexte	non
tolérance au bruit	non
tolérance à l'effet de chaîne	non
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	oui
tolérance aux clusters de forme quelconque	oui
tolérance aux clusters concentriques	oui

TAB. 3.7 – Caractéristiques associées au clustering basé sur les graphes.

La première étape de la méthode consiste donc à calculer les valeurs propres de la matrice de similarité, ainsi que les vecteurs propres associés. Les objets sont alors projetés dans l'espace formé par une partie de ces vecteurs propres associés.

Ensuite, différentes techniques peuvent être utilisées pour partitionner l'ensemble des objets projetés. On peut soit utiliser un algorithme de type K-means dans le nouvel espace, ou bien utiliser un algorithme de partition de graphe pour trouver la zone de coupure maximisant la séparation entre les objets.

Cette technique de partitionnement est alors utilisée pour diviser la base des objets en deux, puis elle est itérée sur chaque sous-groupe ciblé, jusqu'à ce que le nombre de clusters recherchés par l'utilisateur soit atteint. Autrement dit, une telle méthode requiert souvent le nombre de clusters recherchés, mais plusieurs techniques ont été suggérées pour définir automatiquement ce paramètre.

L'intérêt d'utiliser de telles transformations des données est de se placer à chaque étape de la méthode dans un sous-espace dans lequel les différences entre clusters sont plus importantes que dans l'espace initial. Par exemple, utiliser une telle technique avec l'algorithme K-means permet de cibler des clusters d'orientation quelconque, alors que K-means seul n'est pas capable de cibler de tels clusters.

Ce type de méthode ayant émergé relativement récemment, de nombreuses variantes ont été proposées. Puisqu'elles sont capables de transformer l'espace de description des objets, elles sont potentiellement capables d'identifier des clusters de forme quelconque. Par contre, partant d'une matrice de similarité entre toutes les paires d'objets, puis utilisant des méthodes basées sur des calculs matriciels, leur complexité est élevée.

3.9 Clustering hybride

Le développement de certaines méthodes hybrides entre les versions hiérarchiques et les versions par partition du clustering a été motivé par l'exploitation des avantages de ces deux techniques qui semblent complémentaires : le clustering hiérarchique est plus souple que le clustering par partition, alors que ce dernier a une complexité en temps et en espace qui est inférieure.

Ainsi, une méthode de combinaison proposée dans [Steinbach et al., 2000] consiste par exemple à :

1. former un unique cluster regroupant tous les objets ;
2. itérer jusqu'à ce que le nombre désiré de clusters soit atteint :
   - choisir un cluster à diviser selon un certain critère, en favorisant éventuellement ceux qui contiennent davantage d'objets ;
   - trouver deux sous-clusters en utilisant l'algorithme K-means.

Dans [Wang et al., 1997], une autre méthode de combinaison a été proposée. Dans ce cas, la méthode hiérarchique est combinée à une méthode basée sur les grilles. En haut de la hiérarchie, une unique cellule représente l'ensemble de l'espace, puis plus on descend dans la hiérarchie, plus la résolution de la division de l'espace par les grilles est fine.

Dans [Karypis et al., 1999], le principe suivi consiste à créer un ensemble initial important de *petits* clusters en utilisant une méthode par partition du graphe des plus proches voisins, puis à les fusionner en utilisant un algorithme hiérarchique ascendant.

Les caractéristiques des méthodes mises en œuvre dans ce cadre dépendent donc des méthodes utilisées lors de la combinaison, c'est pourquoi nous ne proposons ici aucun tableau récapitulatif de leurs caractéristiques. Nous renvoyons pour cela aux tableaux associés aux méthodes combinées.

3.10 Subspace clustering ascendant

Comme nous l'avons évoqué dans la section 2.3, plusieurs méthodes de clustering ont relativement récemment émergé qui ont pour but de prendre en compte le fait que les clusters peuvent être projetés dans différents sous-espaces de l'espace original de description des objets. On parle dans ce cas de subspace clustering. Dans ce cadre, deux grandes familles de méthodes se distinguent :

1. pour les méthodes *ascendantes sur les dimensions*, on cherche d'abord les sous-espaces les plus appropriés pour la détection de clusters, et on cherche ensuite les clusters présents dans ces sous-espaces ;
2. au contraire, pour les méthodes *ascendantes sur les dimensions*, on cherche d'abord les clusters présents dans les données, puis on cible ensuite les sous-espaces qui leur correspondent le mieux.

Basé sur l'utilisation de grilles, le principe général du subspace clustering ascendant sur les dimensions [Agrawal et al., 1998] est de rechercher les ensembles de cellules denses

connectées dans des sous-espaces de l'espace original. Comme toute méthode basée sur les grilles, il nécessite donc la donnée de deux paramètres utilisateurs : ξ est un entier utilisé pour déterminer la taille de la grille à utiliser, obtenue en partitionnant chaque dimension de l'espace de description des objets en ξ intervalles de même longueur, et τ représente le pourcentage minimum d'objets qui doivent être contenus dans une cellule de la grille pour qu'elle soit considérée comme dense.

La méthode proposée se décompose en trois phases :

1. cibler les sous-espaces dans lesquels se trouvent des clusters denses ;
2. identifier les clusters dans ces sous-espaces ;
3. générer une description minimale des clusters trouvés.

Concernant la première phase, l'algorithme utilise la propriété de monotonie suivante : *si un ensemble d'objets C_k est un cluster dans un sous-espace à m dimensions, alors C_k fait également partie d'un cluster dans toute projection à $(m-1)$ dimensions de ce sous-espace.*

Basée sur ce principe, une approche ascendante est donc utilisée pour cibler les sous-espaces dans lesquels se trouvent des clusters denses. On démarre en projetant les objets sur chaque dimension de l'espace. Pour sélectionner les sous-espaces pertinents à considérer (ceux étant susceptibles de contenir des clusters denses), la technique proposée est de sélectionner les sous-espaces qui maximisent la couverture de l'espace des objets par les cellules denses (mesure qui sera faible si les objets sont uniformément distribués dans le sous-espace, et inversement), en se basant sur le principe MDL (*Minimum Description Length*) pour décider de la zone de coupure entre sous-espaces conservés et sous-espaces supprimés.

Puis, pour passer des sous-espaces candidats S_{m-1} à $(m-1)$ dimensions aux sous-espaces candidats S_m à m dimensions, on combine deux sous-espaces de S_{m-1} à $(m-1)$ dimensions si ceux-ci ont leur $(m-2)$ premières dimensions en commun, et si tout sous-espace à $(m-1)$ dimensions inclus dans ce nouveau sous-espace à m dimensions est déjà inclus dans S_{m-1}.

Ensuite, lorsque l'ensemble des sous-espaces jugés comme pertinents a été identifié et qu'il n'y a plus d'autre sous-espace candidat, l'étape suivante de la méthode consiste à cibler les clusters dans ces sous-espaces, c'est-à-dire à identifier les ensembles de cellules denses connectées en utilisant un clustering basé sur les grilles.

Enfin, la dernière phase de description des clusters se décompose en deux étapes :

1. effectuer une couverture de chaque cluster par des régions maximales ;
2. effectuer une sélection parmi ces régions pour obtenir une couverture minimale.

La première de ces deux phases se fait par un algorithme glouton :
- démarrer avec une cellule sélectionnée aléatoirement ;
- sélectionner une dimension aléatoirement et élargir la région courante sur la droite ou la gauche si la nouvelle région ne couvre pas de cellule non dense, et ce jusqu'à ce que toutes les dimensions aient été envisagées ;
- et ainsi de suite jusqu'à ce que toutes les cellules soient couvertes.

Puis finalement, pour obtenir une couverture minimale, on supprime les régions redondantes en commençant par celles qui couvrent le moins de cellules.

Cette méthode étant basée sur l'utilisation de grilles, ses caractéristiques sont les mêmes que les méthodes basées sur les grilles développées dans l'espace complet de description des données, à l'exception de la prise en compte du contexte qui est réalisée ici. En particulier, les deux paramètres de taille de cellule et de critère de densité étant communs à tous les clusters, ils ne permettent pas d'identifier des clusters ayant des densités très différentes.

3.11 Subspace clustering descendant

La deuxième famille de méthodes de subspace clustering utilise une approche descendante sur les dimensions. Les méthodes développées dans ce cadre proposent des techniques originales de sélection des dimensions pertinentes associées aux groupes d'objets identifiés.

Ainsi, PROCLUS [Aggarwal et al., 1999] et FINDIT [Woo and Lee, 2002] utilisent une approche stochastique basée sur les K médoïdes. Pour associer à chaque cluster l'ensemble des dimensions qui lui correspondent le mieux, PROCLUS demande à l'utilisateur de spécifier le nombre moyen de dimensions pertinentes des clusters et sélectionne celles pour lesquelles la distance moyenne des membres des clusters à leur centre sur ces dimensions est minimale.

Pour FINDIT, les dimensions pertinentes des clusters sont celles sur lesquelles les clusters respectent un critère de densité minimale spécifié par l'utilisateur : pour un médoïde et une dimension donnés, si $\alpha\%$ des k plus proches voisins du médoïde sont distants de moins de ϵ sur la dimension, alors celle-ci est considérée comme caractéristique du cluster associé au médoïde.

DOC [Procopiuc et al., 2002] utilise également une recherche stochastique basée sur deux paramètres α et w. Un cluster est considéré comme pertinent s'il est composé d'au moins $\alpha\%$ des objets de la base, et si l'intervalle de définition de ses membres est inférieur à w sur ses dimensions pertinentes associées. Les clusters conservés sont alors ceux qui maximisent le nombre d'objets ainsi que le nombre de dimensions associées.

Quant à HARP [Yip et al., 2003], constatant qu'il est très difficile pour l'utilisateur de spécifier les paramètres requis par ces différentes méthodes, il propose d'utiliser des seuils dynamiques. Un seuil contrôle le nombre de dimensions pertinentes minimum associé aux clusters, et un autre seuil est utilisé pour sélectionner ces dimensions pertinentes. La méthode utilisée est hiérarchique ascendante, avec modification des seuils à chaque fois que plus aucune fusion de clusters ne respecte les critères.

Enfin, LAC [Domeniconi et al., 2004] permet lui aussi de se passer de paramètres utilisateurs pour sélectionner les dimensions pertinentes des clusters. Pour cela, il suggère de pondérer les dimensions en fonction de la dispersion des membres des clusters sur ces dimensions, au lieu d'en sélectionner un sous-ensemble.

Les différentes approches proposées dans ce cadre ayant des caractéristiques très différentes, nous ne sommes pas en mesure de présenter ici de tableau récapitulatif de

leurs propriétés générales. Globalement, chacune de ces approches peut être rattachée à une méthode de clustering présentée précédemment (PROCLUS, FINDIT et LAC au clustering K-means, DOC au clustering stochastique, HARP au clustering hiérarchique), sauf qu'elles prennent en compte la problématique du contexte dans lequel les clusters sont créés.

3.12 Bilan

Parmi la grande variété de techniques que nous avons présentées, chacune possède ses forces et ses faiblesses. Le clustering hiérarchique, sous sa forme ascendante et *single-link* est souvent utilisé lorsque les jeux de données sont de petite taille, puisque dans de tels cas la complexité élevée de la méthode n'est pas une limitation. Le clustering basé sur les graphes peut également être utilisé dans de tels cas. Si au contraire des problèmes de temps d'exécution se posent, alors c'est souvent le clustering K-means qui est utilisé. Lorsque les clusters peuvent être de forme allongée, alors le clustering statistique est plus approprié que le clustering K-means. Lorsque les problèmes contiennent encore davantage d'objets, alors le clustering stochastique peut être utilisé. Si l'objectif est de cibler des clusters qui peuvent être de forme quelconque, alors ce sont les clusterings basés sur la densité, sur les grilles ou le clustering spectral qui sont utilisés. Enfin, si les clusters peuvent être définis dans des sous-espaces de l'espace original de description des objets, alors c'est le subspace clustering qui doit être utilisé. C'est donc l'application visée qui va déterminer la méthode de clustering qui doit être utilisée.

Malgré le nombre important de méthodes existantes, plusieurs problématiques restent encore ouvertes dans le cadre du clustering. Un problème très souvent rencontré concerne la difficulté de fixer les paramètres en entrée des méthodes par l'utilisateur. De même, la présentation des résultats sous une forme facilement interprétable par l'utilisateur est rarement abordée malgré son intérêt. La complexité des méthodes mises en œuvre est aussi problématique dans certains cas. La prise en compte du contexte dans lequel les clusters sont créés constitue un sujet d'étude récent, qui mérite donc d'être approfondi. Enfin, une problématique ouverte importante dans le cadre du clustering concerne l'évaluation des résultats et la comparaison de différentes méthodes. Dans la suite de cette thèse, nous présentons donc nos contributions sur ces différents points. La partie II présente ainsi nos travaux dans la mise en œuvre de nouvelles méthodes de subspace clustering, minimisant le nombre de paramètres requis de la part de l'utilisateur, fournissant en sortie un résultat compréhensible, et ayant une complexité raisonnable. La partie III présente ensuite nos travaux dans le domaine de l'évaluation des résultats d'algorithmes de clustering, ou de la comparaison de tels algorithmes.

Deuxième partie

Nouvelles méthodes de clustering

Dans les travaux de recherche que nous avons menés, nous avons abordé plusieurs problématiques ouvertes dans le cadre du clustering.

Tout d'abord, nous nous sommes attaqué au problème de la prise en compte du contexte dans lequel les clusters présents dans les données sont créés, c'est-à-dire aux cas où les clusters peuvent être définis dans différents sous-espaces de l'espace original de description des objets. Autrement dit, nos contributions se situent dans le cadre du subspace clustering.

Par ailleurs, nous nous plaçons dans l'optique d'utiliser le clustering dans un cadre d'extraction de connaissances, l'objectif étant alors d'aider à la compréhension de la structure des données et de faire ainsi émerger certains sous-concepts présents dans les données. Or dans un tel cadre, l'utilisateur a rarement de connaissances a priori sur la structure des données, et la description des résultats sous forme compréhensible s'avère d'une aide précieuse pour l'aider à appréhender au mieux les sous-concepts présents dans les données. Dans la mise en œuvre de nos méthodes, nous nous sommes donc donné comme objectif de minimiser le nombre de connaissances a priori requises de la part de l'utilisateur, et nous nous sommes intéressé à la présentation des résultats sous forme compréhensible et visuelle pour l'utilisateur.

Nous avons également considéré la possibilité d'être confronté à des bases de données hétérogènes regroupant différents types d'attributs, étant donné que dans la plupart des bases de données réelles, les données sont décrites par des attributs numériques aussi bien que catégoriels. Une partie de nos travaux porte par ailleurs sur la mise en œuvre d'une méthode de subspace clustering adaptée aux cas des données semi-structurées. Enfin, nous avons également envisagé la possibilité d'être confronté à de larges bases de données pouvant contenir des données manquantes ainsi que du bruit.

Le chapitre 4 présente ainsi la première méthode de clustering que nous avons développée dans ce cadre, appelée **Tuareg**. Son principe général consiste à repérer à chaque étape la dimension permettant le partitionnement le plus pertinent pour le groupe d'objets considéré, et à itérer ce processus jusqu'à ce que plus aucun groupe ne soit amélioré par découpage, à la manière de C4.5 [Quinlan, 1993] en apprentissage supervisé.

Le chapitre 5 présente ensuite une deuxième méthode de subspace clustering appelée **SuSE** et répondant à certaines limitations observées de **Tuareg**. Nous montrons alors en particulier l'intérêt d'utiliser des modèles probabilistes dans le cadre du subspace clustering.

Puis dans le chapitre 6, nous présentons plusieurs extensions possibles de ces méthodes, dans le cadre d'un apprentissage autre que non supervisé, et face à des données semi-structurées.

Enfin, le chapitre 7 présente les différentes expérimentations que nous avons menées avec ces différentes approches, comparées à d'autres. Ces expérimentations sont d'abord conduites face à des données artificielles, pour observer leur robustesse dans différentes situations contrôlées, puis nous montrons les résultats obtenus face à des données réelles et des données semi-structurées, illustrant l'intérêt de ces approches dans le cas d'applications réelles.

Chapitre 4

Tuareg : divisions successives sur les dimensions de l'espace

Nous proposons ici **Tuareg** (contex**TUA**lized clust**ER**in**G**) [Candillier et al., 2004], un algorithme original s'apparentant à la famille des méthodes de subspace clustering. Dans la mise en œuvre de cet algorithme, nous nous sommes donné pour objectif de minimiser le nombre de connaissances a priori requises de la part de l'utilisateur, et de fournir en sortie un résultat facilement interprétable. Nous nous plaçons ainsi dans le cadre de l'utilisation du clustering pour l'extraction de connaissances.

Dans la première section de ce chapitre, nous discutons d'abord des différentes manières possibles de présenter les résultats d'un clustering sous forme compréhensible pour l'utilisateur, et montrons l'intérêt d'utiliser des règles (hyperrectangles dans des sous-espaces de l'espace original de description des objets) dans ce cadre. En particulier, nous montrons comment cette représentation nous permet de décrire chaque cluster par un minimum d'attributs pertinents.

Le principe général de **Tuareg** consiste à fractionner successivement l'ensemble des objets. Dans l'esprit, l'approche est comparable à celle de C4.5 [Quinlan, 1993] en apprentissage supervisé : il s'agit de repérer à chaque étape la dimension permettant le partitionnement le plus pertinent pour le groupe d'objets considéré, et d'itérer ce processus jusqu'à ce que plus aucun groupe ne soit amélioré par découpage.

Chaque dimension de description est considérée l'une après l'autre et indépendamment des autres. La brique de base est donc un algorithme de partitionnement sur une dimension. C'est ce sujet que nous traitons dans la section 4.2, avant d'exposer comment ce partitionnement élémentaire est intégré dans une stratégie globale visant à identifier, pour chaque groupe, ses dimensions pertinentes, et comment un regroupement complet des objets est finalement constitué.

4.1 Présentation des résultats par les règles

Il existe trois principales techniques pour présenter les résultats d'un clustering sous forme compréhensible pour un utilisateur :

1. projeter les données dans un espace visuel à deux ou trois dimensions ;
2. présenter les résultats à l'aide d'un arbre de décision ;
3. ou présenter les résultats à l'aide de règles.

Dans le premier cas, une visualisation en deux ou trois dimensions des objets permet d'observer aisément la proximité ou l'éloignement entre objets, mais cela nécessite d'avoir un identifiant compréhensible de chacun de ces objets. Par exemple, si les objets représentent des animaux, alors le fait qu'un lion soit plus proche d'un chat que d'un chien apporte de l'information. De même, si les objets sont des documents titrés, alors le fait qu'un document sur l'*apprentissage automatique* soit plus proche d'un document sur l'*informatique* que d'un document sur l'*apprentissage à l'école* apporte de l'information. Par contre, si aucun identifiant compréhensible n'est associé aux objets, alors la proximité observée entre deux objets n'apportera aucune information. Les deux techniques les plus utilisées pour effectuer de telles projections sont les *cartes auto-organisatrices de Kohonen* [Su and Chang, 2000] et l'*Analyse en Composantes Principales* [Jolliffe, 1986], détaillée en annexes B.1.2.

Les arbres de décision sont des structures souvent utilisées pour fournir un résultat compréhensible à l'utilisateur. Or les règles ont un potentiel expressif plus important que les arbres de décision. Si nous considérons le cas présenté par la figure 4.1 par exemple, les trois groupes présents dans les données seront naturellement définis comme suit par un utilisateur :

- le groupe C_0 se situe à droite ;
- le groupe C_1 se situe en haut ;
- et le groupe C_2 se situe en bas à gauche.

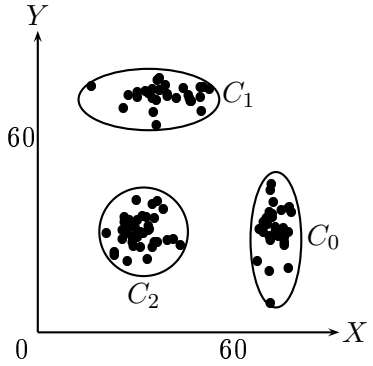


FIG. 4.1 – Exemple de jeu de données.

La formalisation de cette représentation à l'aide de règles est alors directe :

- le cluster C_0 est défini par des valeurs supérieures à 60 sur X :

$$C_0 : X \in [60, 80]$$

- le cluster C_1 est défini par des valeurs supérieures à 60 sur Y :

$$C_1 : Y \in [60, 80]$$

– et le cluster C_2 est défini par des valeurs inférieures à 60 sur X et sur Y :

$$C_2 : X \in [20, 40] \text{ et } Y \in [20, 40]$$

Par contre, un arbre de décision obtenu dans un tel cas pourrait être celui présenté par la figure 4.2. Or si l'on observe comment est caractérisé le cluster C_1 dans ce cas, on peut remarquer qu'il est défini par des valeurs supérieures à 60 sur X et sur Y, alors qu'il n'est pas utile de considérer les valeurs des membres de C_1 sur X pour caractériser leur appartenance au cluster. Dans un tel cas, l'utilisation de règles permet donc de minimiser le nombre d'attributs nécessaires pour caractériser les clusters.

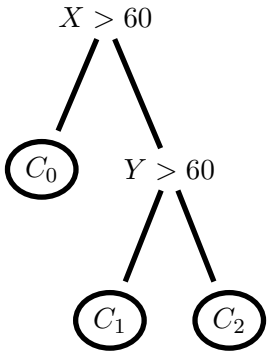


FIG. 4.2 – Exemple d'arbre de décision obtenu sur le jeu de données.

Par ailleurs, dans des cas comme ceux qui sont présentés dans la figure 4.3, la description des clusters par des règles est plus appropriée que celle utilisant les arbres de décision car les règles permettent naturellement la prise en compte des chevauchements entre clusters, alors que les arbres de décision ne pourront faire autrement que de découper l'un des clusters en deux. En effet, dans les deux cas, le premier test de l'arbre de décision devra porter sur un découpage en 1 ou 2 sur l'un des deux axes, ce qui aura fatalement pour effet de diviser l'un des deux clusters.

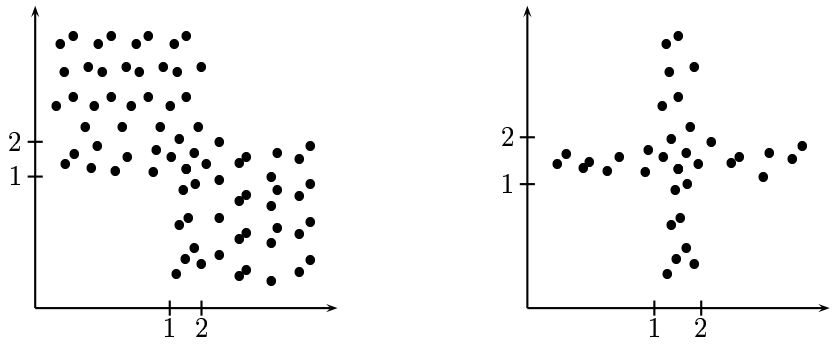


FIG. 4.3 – Intérêt de l'utilisation de règles pour représenter les clusters.

Utiliser des hyperrectangles plutôt que des hyperellipsoïdes pour représenter les clusters permet par ailleurs à l'utilisateur d'interpréter plus facilement les résultats. De

plus, calculer l'appartenance des objets aux clusters lorsque ceux-ci sont représentés par des hyperrectangles est moins coûteux que lorsque des hyperellipsoïdes sont utilisés, ce qui permet en particulier, nous le verrons dans la suite, de simplifier plus naturellement leur description en réduisant le nombre d'attributs considérés.

On représente aussi souvent les clusters par leurs centroïdes, mais dans de nombreux cas, les différences entre clusters ne sont ainsi pas mises en avant, surtout lorsque le nombre d'attributs décrivant ces centroïdes est important. De plus, si dans le premier cas de la figure 4.3, les différences entres clusters peuvent être mises en avant par la présentation des centres de clusters, elles ne le seront pas du tout dans le deuxième cas pour lequel les deux centres se superposent.

Dans [Torre, 1999] et [Sarafis et al., 2003], des méthodes sont proposées pour rechercher des ensembles de règles pour représenter les données. La première est basée sur une création ascendante de ces règles alors que la seconde utilise un algorithme génétique pour leur formation. Cependant, ces deux méthodes sont basées sur l'utilisation d'un seuil de densité des règles produites qui doit être fourni par l'utilisateur. Or de tels paramètres sont souvent difficiles à fournir, et les résultats sont dépendants des valeurs choisies. De plus, de telles méthodes basées sur l'utilisation d'un seuil de densité défini de manière globale ne sont pas capables de cibler des clusters ayant des densités très différentes.

Finalement, pour la mise en œuvre de notre méthode, nous avons donc choisi de nous baser sur l'utilisation de règles pour représenter les clusters identifiés, et de minimiser le nombre de connaissances a priori requises de la part de l'utilisateur.

4.2 Partitionnement sur une dimension

La brique de base de notre méthode est un algorithme de partitionnement sur une dimension, qui sera utilisé sur n'importe quel sous-ensemble d'objets et n'importe quelle dimension de façon itérative.

On peut à ce niveau envisager de nombreuses méthodes pour partitionner un ensemble E de n valeurs sur une seule dimension $\{v_1, ..., v_n\}$. Dans le cas d'une dimension numérique, une première approche possible consiste à diviser cet ensemble en deux en coupant entre les deux valeurs consécutives séparées par la plus grande distance. Si les valeurs v_i sont rangées par ordre croissant, cela revient donc à chercher :

$$Dist_{max}(E) = Max_{i=1}^{n-1}(v_{i+1} - v_i)$$

Cette technique est alors tout à fait efficace lorsque des séparations importantes entre les données existent sur certaines dimensions, et s'il n'y a pas de bruit dans les données. C'est le cas dans l'exemple de la figure 4.4, où un ensemble de données projetées sur une dimension suivent deux distributions de probabilités de densité bien séparées.

Par contre, la présence de bruit dans les données risque de mettre cette méthode en défaut à cause de l'apparition d'objets bruités dans les zones de séparation des groupes. De la même manière, si les données ne sont que faiblement séparées lors de leur projection sur une dimension, alors la méthode peut également échouer à cibler la

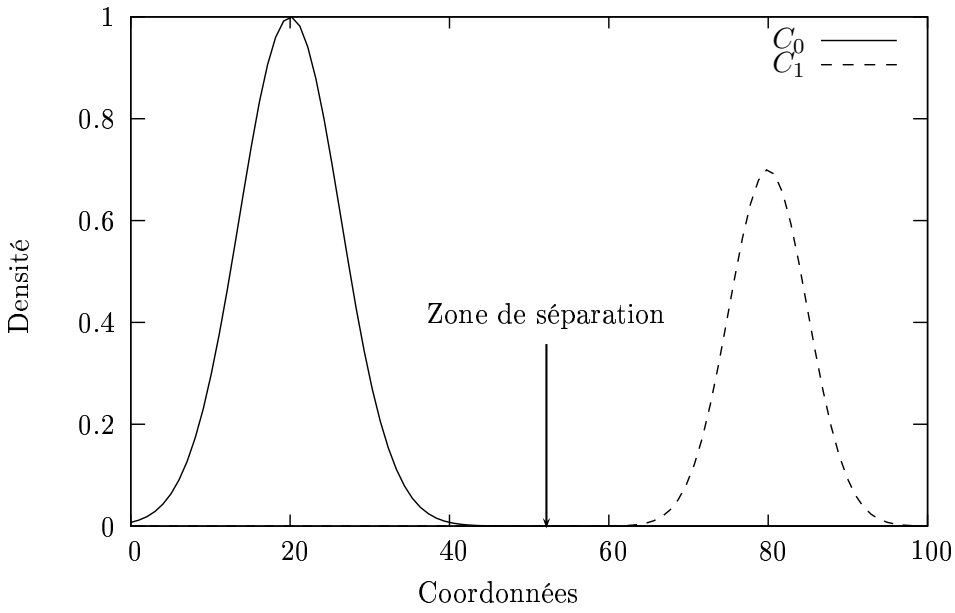


FIG. 4.4 – Exemple de séparation des données projetées sur une dimension.

meilleure zone de séparation, même s'il est plus probable que les valeurs consécutives séparées par la plus grande distance se situent dans les zones de séparation des groupes plutôt qu'en leurs centres. Un tel cas est illustré par la figure 4.5.

Une solution pour pallier à cette limitation consiste à rechercher la distance maximale entre deux objets contenus dans un certain voisinage, au lieu de chercher à maximiser la distance entre objets directement voisins. On améliore ainsi la probabilité de cibler des objets proches des zones de séparation des groupes. Dans ce cas, si l'on pose r le nombre de voisins à considérer (égal à 5% de n par exemple), cela revient donc à chercher :

$$Dist_{max}(E) = Max_{i=1}^{n-r}(v_{i+r} - v_i)$$

La zone de séparation correspondante serait alors située au milieu des deux objets séparés par la plus grande distance, c'est-à-dire à la valeur suivante :

$$\frac{v_{j+r} + v_j}{2} \text{ pour } j = ArgMax_{i=1}^{n-r}(v_{i+r} - v_i)$$

Par ailleurs, d'autres méthodes plus complexes de partitionnement sur une dimension peuvent être considérées. La méthode de Fisher [Diday et al., 1982a], détaillée en annexes C.1, ou toute méthode générale de clustering, telle la méthode de clustering K-means ou de clustering statistique, peuvent être envisagées.

Enfin, on peut choisir de diviser les données en deux groupes, quitte à rediviser ensuite les données sur la même dimension puisque cette étape sera itérée, ou alors considérer différents nombres de groupes possibles lors des découpages sur une dimension.

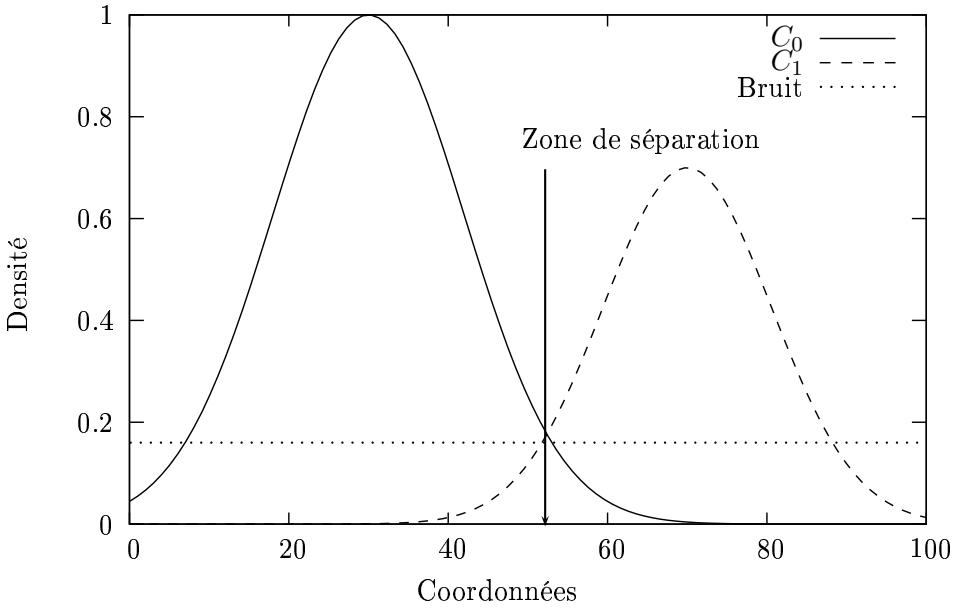


FIG. 4.5 – Exemple de données bruitées et faiblement séparées.

L'ensemble de ces choix influence évidemment le temps d'exécution de la méthode. Or il s'agit d'une méthode élémentaire qui sera exécutée de nombreuses fois pendant l'algorithme global. La complexité de cette opération élémentaire va donc déterminer la complexité générale de la méthode. C'est pourquoi dans la suite, nous considérons comme opération élémentaire de partitionnement sur une dimension celle qui consiste à diviser en deux entre les deux valeurs contenues dans un certain voisinage et séparées par la plus grande distance.

4.3 Évaluer l'intérêt d'une division

À ce niveau, nous considérons que nous disposons d'un algorithme de partitionnement sur une dimension que nous nommons *Clustering1D*, qui admet en entrée un ensemble de valeurs et qui fournit en sortie une partition en deux de cet ensemble.

Cet algorithme peut être utilisé sur n'importe quel cluster déjà formé en projetant l'ensemble des objets qui lui est associé sur n'importe quelle dimension. L'étape suivante consiste alors à choisir la meilleure division parmi toutes les candidates. Pour cela, deux considérations sont à prendre en compte :

1. est-ce que la séparation entre les deux nouveaux clusters est significative ?
2. les nouveaux clusters formés sont-ils plus intéressants que le cluster de départ ?

On retrouve ici le compromis classique entre la maximisation des distances inter-clusters (critère 1) et la minimisation des distances intra-clusters (critère 2), mais simplifié dans notre cas puisque dans l'évaluation du premier critère, les distances sont

ramenées à une seule dimension, et que seul un sous-ensemble des dimensions sélectionnées jusqu'alors est considéré pour l'évaluation du second critère.

Soit $P_d(C_k) = \{\pi_1, \pi_2\}$ la partition obtenue par la méthode *Clustering1D* à laquelle on a fourni en entrée l'ensemble D_{kd} des coordonnées sur la dimension d des objets appartenant au cluster initial C_k. π_1 et π_2 sont les deux intervalles correspondant créés sur la dimension d.

Pour évaluer le premier critère, la mesure utilisée est tout simplement la distance de séparation entre les deux intervalles formés $Dist_{max}(D_{kd})$, normalisée par $|D_{kd}|$, la taille de l'intervalle de définition des coordonnées des objets de C_k sur la dimension d (on considère l'espace dans lequel C_k se trouve, et non pas l'espace complet de description S), pour permettre de comparer des divisions sur des dimensions de tailles différentes, et pondérée par le nombre d'objets du cluster initial C_k pour favoriser la division des clusters contenant davantage d'objets :

$$Interet(P_d(C_k)) = N_k \times \frac{Dist_{max}(D_{kd})}{|D_{kd}|}$$

Plus cette quantité est grande, plus la division correspondante est jugée intéressante. On cherche bien ainsi à maximiser la distance séparant les nouveaux clusters produits.

Encore une fois, à ce niveau, différentes mesures de la qualité interne d'un partitionnement sur une dimension peuvent être envisagées, mais il faut qu'elles soient en accord avec la méthode de partitionnement utilisée. Si c'est un clustering statistique qui est utilisé pour partitionner les données en deux, alors le critère de la vraisemblance du modèle par rapport aux données est tout à fait approprié. Si différents partitionnements avec différents nombres de groupes recherchés sont considérés, alors le critère BIC peut être utilisé.

L'idée du second critère est de refuser une division si celle-ci marque une baisse de l'homogénéité du cluster sur ses dimensions caractéristiques. Cette homogénéité est évaluée par l'inertie moyenne du cluster sur ses dimensions caractéristiques S_k :

$$Inertie(C_k) = \frac{\sum_{\{d \in S_k\}} Inertie_d(C_k)}{|S_k|}$$

$$Inertie_d(C_k) = \frac{\sum_{\{i \in D_k\}} (x_{id} - \mu_{kd})^2}{|D_{kd}|}$$

$$\mu_{kd} = \frac{\sum_{\{i \in D_k\}} x_{id}}{N_k}$$

Plus l'inertie est grande, plus les membres du cluster considéré sont dispersés, et donc moins le cluster est intéressant. On cherche donc bien ainsi à minimiser les distances internes des nouveaux clusters produits.

Nous nommons *AcceptDivision* la fonction retournant s'il faut accepter ou non une division donnée $P_d(C_k)$, ses différentes étapes étant détaillées dans l'algorithme 1.

Ainsi, on choisit de ne considérer que les divisions pour lesquelles l'inertie maximale sur la dimension de division est inférieure à l'inertie du cluster initial considéré, ce qui

Algorithme 1 AcceptDivision

Entrée : $P_d(C_k) = \{\pi_1, \pi_2\}$ la division proposée du cluster C_k sur la dimension d.
calculer $Inertie(C_k)$, l'inertie de C_k sur S_k
for $i = 1$ to 2 **do**
$\mu_i = \frac{\sum_{\{j \in D_k | x_{jd} \in \pi_i\}} x_{jd}}{|\{j \in D_k | x_{jd} \in \pi_i\}|}$ {centre de gravité de D_{kd} sur π_i}
$I_i = \sum_{\{j \in D_k | x_{jd} \in \pi_i\}} (x_{jd} - \mu_i)^2$ {inertie de D_{kd} sur π_i}
end for
$maxInertie = \frac{Max(I_1, I_2)}{|D_{kd}|}$ {inertie normalisée maximale entre π_1 et π_2}
return $(maxInertie < Inertie(C_k))$
Sortie : 1 si la division est acceptée, 0 sinon.

revient à n'accepter une division que si les nouveaux clusters réduisent tous les deux leur inertie. On peut aussi choisir de ne considérer que les divisions pour lesquelles la plus petite inertie sur la dimension de division est inférieure à l'inertie du cluster initial considéré, ce qui revient à remplacer Max par Min dans l'algorithme 1, et à accepter une division si au moins l'un des deux nouveaux clusters générés diminue son inertie. Ainsi, si on utilise l'inertie minimale, alors les divisions sont plus facilement acceptées, et au contraire, en utilisant l'inertie maximale, le nombre de divisions est réduit. Nous verrons plus en détails dans le chapitre 7 présentant nos expérimentations les différences que ce choix entraîne quant au nombre de clusters finaux générés.

Finalement, la méthode de sélection de la division suivante à considérer, étant donnée la partition courante, est nommée *SelectDivision*, et ses étapes principales sont décrites par l'algorithme 2.

Algorithme 2 SelectDivision

Entrée : $P = (C_1, ..., C_K)$ une partition composée de K clusters.
$Best = (0, 0, \emptyset)$ {la meilleure division courante}
$BestNote = 0$ {la mesure d'intérêt associée à la meilleure division}
for $k = 1$ to K **do**
for $d = 1$ to M **do**
$\pi = Clustering1D(D_{kd})$; $Note = Interet(\pi)$
if $(Note > BestNote)$ **and** $(AcceptDivision(\pi))$ **then**
$BestNote = Note$; $Best = (k, d, \pi)$
end if
end for
end for
Sortie : $Best$: k le numéro du cluster à diviser, d la dimension sur laquelle effectuer la division, et $\pi = P_d(C_k)$ la division à effectuer sur cette dimension.

4.4 Algorithme de base : divisions successives

À ce niveau, nous disposons d'un algorithme qui, étant donnée une partition courante, désigne la division suivante à effectuer ou indique que plus aucune division n'est pertinente. Nous en déduisons naturellement la méthode de génération de nouveaux clusters à partir d'une division sélectionnée $P_d(C_k) = \{\pi_1, \pi_2\}$ du cluster C_k sur la dimension d. Soit $SubCluster(C_k, d, \pi_i)$ la fonction qui retourne le cluster C_l :

- dont la vue D_l sur les objets de D inclut les éléments de D_k dont les coordonnées sur la dimension d appartiennent à l'intervalle π_i :

$$D_l = \{j \in D_k | x_{jd} \in \pi_i\}$$

- et dont la deuxième vue sur les dimensions de S comprend les dimensions caractéristiques de C_k plus la dimension d :

$$S_l = S_k \cup d$$

  Mais il peut arriver qu'en ajoutant à un cluster C_l une nouvelle dimension caractéristique, une autre dimension caractéristique de ce cluster $d' \in S_l$ ne soit plus pertinente : ainsi, s'il n'existe aucun objet $\overrightarrow{x_j} \in D \setminus D_l$ tel que pour toute dimension $e \in S_l \setminus d'$, x_{je} appartient à D_{le}, alors on supprime la dimension caractéristique d' de S_l. On obtient ainsi une *description minimale* du cluster C_l, en ciblant l'ensemble minimal des dimensions supports des intervalles qui caractérisent l'appartenance d'un objet à ce cluster. Cette phase de description minimale est illustrée par la figure 4.6.

Finalement, la méthode itérative de base, nommée *Divisif*, est présentée par l'algorithme 3. Le seul paramètre de cet algorithme est Max_K et correspond au nombre maximum de clusters générés. Il ne s'agit pas du nombre attendu de clusters mais d'une borne supérieure qui n'est pas nécessairement atteinte. L'influence de ce paramètre sur les résultats est discuté dans les expérimentations présentées dans le chapitre 7. Le résultat de cet algorithme sur un exemple élémentaire est donné par la figure 4.6.

4.5 Algorithme final : Tuareg

La version de base de l'algorithme *SelectDivision* se contente de rechercher le meilleur découpage possible sur une dimension de l'un des clusters courants. Or il est souvent plus pertinent de conserver les k meilleurs choix possibles lors d'une recherche descendante. C'est le principe de la *recherche en faisceau*, ou *beam search*. De plus, nous voulons introduire une certaine souplesse dans l'algorithme, en ne requérant pas qu'il effectue un partitionnement strict de l'ensemble des données de départ, mais qu'il constitue plutôt un ensemble de clusters possibles homogènes qui peuvent éventuellement avoir des intersections non vides.

Pour satisfaire ces deux exigences tout en conservant un algorithme efficace, nous proposons d'introduire une dimension stochastique dans l'algorithme *SelectDivision*. L'idée est de remplacer la sélection du meilleur découpage possible à chaque étape par

Algorithme 3 Divisif

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, et Max_K, borne supérieure sur le nombre de clusters générés.
$K = 1$; $P[1] = (D, \emptyset)$ {P la partition courante composée de K clusters et initialisée avec un cluster contenant tous les objets de D et aucune dimension caractéristique}
$(k, d, \pi) = SelectDivision(P)$
while $(K < Max_K)$ **and** $(\pi \neq \emptyset)$ **do**
$K = K + 1$
$P[K] = SubCluster(P[k], d, \pi_1)$
$P[k] = SubCluster(P[k], d, \pi_2)$
$(k, d, \pi) = SelectDivision(P)$
end while
Sortie : P la partition créée de D, composée de K clusters.

un *tirage roulette* (tirage au sort avec des probabilités proportionnelles à leur mesure d'intérêt) parmi les k (nommé *ALEA_DIV* par la suite) meilleurs découpages possibles. Cette idée conduit à remplacer l'algorithme 2 par l'algorithme 4.

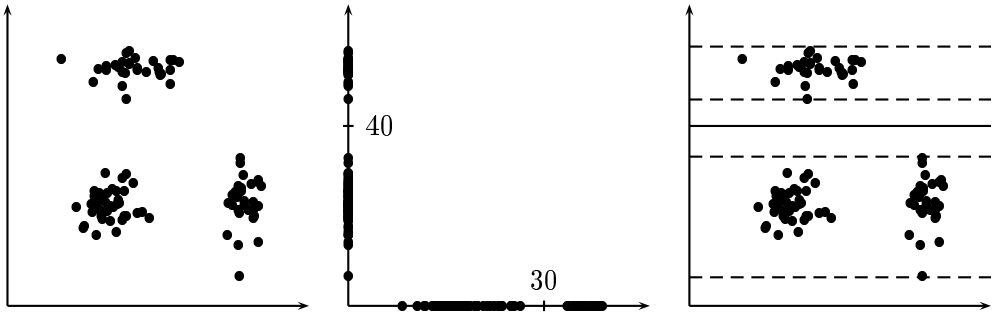
De manière générale, la dimension stochastique de la méthode a pour but de limiter les biais possibles dûs à la simplicité de notre algorithme de partitionnement sur une dimension et dûs aux choix des mesures d'intérêts effectués. Elle permet ainsi de simuler, pour un coût algorithmique raisonnable, la recherche parallèle de solutions alternatives à chaque fois qu'un choix de partitionnement est effectué. Après plusieurs exécutions de l'algorithme de base, il s'agit ensuite de sélectionner parmi tous les clusters générés ceux qui sont les plus denses sur leurs dimensions caractéristiques et couvrent le plus d'objets non encore couverts. La fonction de sélection correspondante est nommée *FinalSelect* et formalisée par l'algorithme 5.

Finalement, **Tuareg**, notre algorithme de *clustering contextualisé*, est décrit par l'algorithme 6 et consiste à :

1. exécuter l'algorithme de base *Divisif* un certain nombre de fois (*NB_RUNS*), en faisant un tirage aléatoire parmi les meilleures divisions proposées ;
2. sélectionner dans l'ensemble des clusters résultants le sous-ensemble de ceux qui sont les plus denses sur leurs dimensions caractéristiques et couvrent l'ensemble des objets de la base (algorithme 5).

Il nécessite la donnée de trois paramètres en entrée : *ALEA_DIV* et *NB_RUNS*, utilisés pour gérer la composante stochastique de notre algorithme, et Max_K, borne supérieure sur le nombre total de clusters générés pour une itération de la méthode. Nous verrons dans le chapitre 7 l'influence de ces différents paramètres sur les résultats.

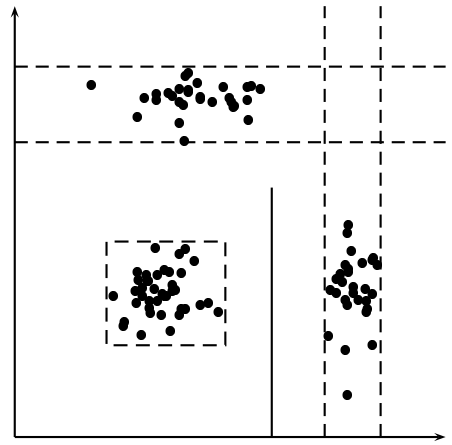
La complexité dans le pire des cas de **Tuareg** est en $O(M \times N \log N)$. La complexité en $O(N \log N)$ selon le nombre d'objets N provient de l'algorithme *Clustering1D* qui nécessite l'ordonnancement des valeurs sur une dimension pour cibler la distance maximale entre valeurs contenues dans un certain voisinage. La complexité linéaire en fonction du nombre de dimensions M provient, elle, de l'algorithme *SelectDivision*, qui néces-



(a) Base d'exemples.

(b) Phase 1 : *Clustering1D* sur chaque dimension et mesures d'intérêt associées.

(c) Phase 2 : division sur la dimension d'intérêt maximum.



(d) Itérations : divisions tant que possible, et description minimale des clusters.

FIG. 4.6 – Exemple d'exécution de l'algorithme *Divisif*.

site d'exécuter l'algorithme *Clustering1D* sur chaque dimension de l'espace. Quant aux autres composantes de l'algorithme, leur complexité est inférieure, car bornée par des constantes dépendantes des paramètres d'entrée de la méthode, dont les valeurs sont supposées négligeables devant M et N. Plus précisément, la méthode est linéaire en fonction des paramètres d'entrée $ALEA\_DIV$, NB_RUNS et Max_K.

4.6 Bilan

Tuareg se distingue de ses concurrents par un certain nombre de caractéristiques. La principale est qu'il produit un ensemble de clusters hyperrectangulaires définis par des intervalles sur un *nombre restreint de dimensions caractéristiques*. Il ne réalise pas une partition de l'ensemble des données puisque les clusters finaux peuvent se chevaucher partiellement.

Algorithme 4 SelectDivision (version 2, stochastique)

Entrée : $P = (C_1, ..., C_K)$ une partition composée de K clusters, et $ALEA\_DIV$, contrôlant la composante stochastique de la méthode.
$Bests[1..ALEA\_DIV] \leftarrow$ initialiser à $(0, 0, \emptyset, 0)$ {tableau stockant les $ALEA\_DIV$ divisions dont la mesure d'intérêt associée est maximale}
$MinBest = 0$ {note minimale associée aux divisions stockées dans $Bests$}
for $k = 1$ to K **do**
  for $d = 1$ to M **do**
    $\pi = Clustering1D(D_{kd})$; $Note = Interet(\pi)$
    if $(Note > MinBest)$ **then**
      Cibler l'indice i de l'emplacement, dans $Bests$, de la division de note minimale
      $Bests[i] = (k, d, \pi, Note)$; Mettre à jour $MinBest$
    end if
  end for
end for
for $i = 1$ to $ALEA\_DIV$ **do**
  if $(Not\ AcceptDivision(Bests[i]))$ **then**
    Supprimer le i^{eme} élément du tableau $Bests$
  end if
end for
Retourner aléatoirement l'une des divisions stockées dans $Bests$, avec la probabilité de tirage d'une division proportionnelle à sa mesure d'intérêt associée

Sortie : k le numéro du cluster à diviser, d la dimension sur laquelle effectuer la division, et $\pi = P_d(C_k)$ la division à effectuer sur cette dimension.

L'hypothèse fondamentale sur laquelle est basé **Tuareg** est que pour tout couple de clusters distincts, il existe toujours au moins une dimension de projection sur laquelle ces deux clusters sont bien séparés. Cette hypothèse est tout à fait comparable à celle qui fonde l'efficacité de C4.5 en apprentissage supervisé.

De par ces caractéristiques, **Tuareg** ne peut traiter correctement des cas comme ceux de la figure 4.7. Dans tous ces exemples, même quand les clusters sont bien séparés dans l'espace complet de description, cette séparation n'est visible sur aucune projection des objets sur une seule dimension, ce qui fait échouer l'algorithme de partitionnement élémentaire.

Pour relâcher la contrainte que les séparations entre clusters doivent être visibles lors de la projection des objets sur une unique dimension de description (ce qui signifie qu'elles doivent être parallèles aux axes), il est possible d'ajouter à la méthode une première phase d'*Analyse en Composantes Principales* (méthode présentée en annexes B.1.2). Cette phase devrait permettre, via un changement de repère, de retrouver des clusters dont l'orientation générale est parallèle aux axes, alors que ce n'était pas le cas initialement. Cette phase préliminaire risque néanmoins d'entraîner des temps de calculs plus conséquents puisqu'une telle analyse a une complexité quadratique en fonction du nombre de dimensions, et elle rend alors les résultats moins facilement

Algorithme 5 FinalSelect

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, et SET, un ensemble de clusters candidats pouvant se recouvrir.
$NbUncover = N$ {nombre d'objets non couverts par les clusters sélectionnés}
$Uncover = \{\overrightarrow{x_1}, ..., \overrightarrow{x_N}\}$ {liste des objets non couverts}
$P = \emptyset$ {la partition résultante}
while $(NbUncover > 0)$ **do**
  $Best = \emptyset$ {cluster courant le plus dense}
  $BestDensite = 0$ {meilleure densité courante}
  $NbCover = 0$ {nombre d'objets de $Best$ non couverts}
  for all $C_k \in SET$ **do**
    $Ncover = |D_k \cap Uncover|$ {nombre d'objets de C_k non couverts}
    $Volume = \sqrt[|S_k|]{\prod_{d \in S_k} |D_{kd}|}$ {moyenne géométrique des tailles d'intervalles de définition de C_k sur ses dimensions caractéristiques, la moyenne étant utilisée pour permettre que les volumes de clusters définis sur différents nombres de dimensions caractéristiques soient comparables}
    $Densite = Ncover/Volume$
    if $(Densite > BestDensite)$ **then**
      $Best = C_k$; $BestDensite = Densite$; $NbCover = Ncover$
    end if
  end for
  $P = P \cup Best$; $SET = SET \setminus Best$
  $NbUncover = NbUncover - NbCover$; $Uncover = Uncover \setminus Best$
end while
for all $C_k \in P$ **do**
  for all $C_l \in P$ **do**
    if $D_k \subset D_l$ **then**
      Supprimer C_k de P {cluster inclus dans un autre}
    end if
  end for
end for
Sortie : $P = (C_1, ..., C_K)$ le clustering résultat, composé de K clusters.

Algorithme 6 Tuareg

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, $ALEA\_DIV$ contrôlant la composante stochastique de la méthode, NB_RUNS le nombre d'itérations de la méthode stochastique, et Max_K la borne supérieure sur le nombre de clusters générés pour une itération de la méthode.

$SET = \emptyset$

for $n = 1$ to NB_RUNS **do**

$SET = SET \cup Divisif(D, ALEA\_DIV, Max_K)$

end for

return $FinalSelect(D, SET)$

Sortie : $P = (C_1, ..., C_K)$ le clustering résultat, composé de K clusters.

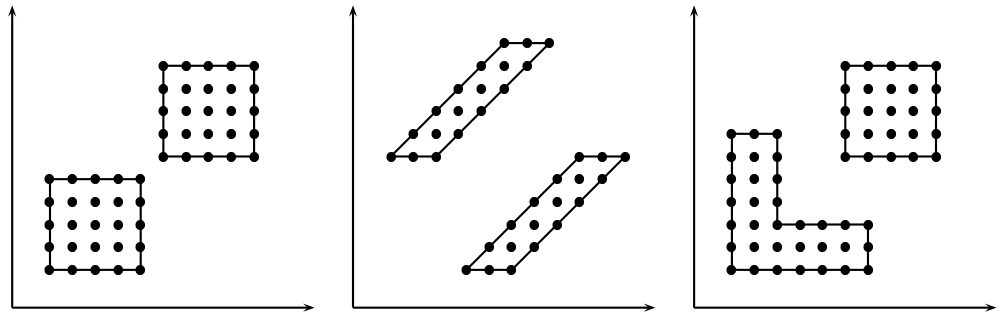


FIG. 4.7 – Exemples de cas problématiques pour **Tuareg**.

interprétables. Une autre possibilité est de considérer des projections sur plus d'une dimension, mais la complexité de la méthode serait alors fortement augmentée. De plus, la spécification de la dimensionalité maximale de projection à considérer serait alors une problématique supplémentaire à gérer, et on pourrait alors souhaiter qu'elle soit égale à la dimensionalité du problème. Enfin, l'esprit de la méthode serait ainsi perdu.

Par ailleurs, l'un des principaux intérêts de **Tuareg** est qu'il demande très peu de *connaissance a priori* de la part de l'utilisateur, contrairement à de nombreuses méthodes qui requièrent de spécifier le nombre exact de clusters recherchés, ou leur inertie maximale par exemple. Nous verrons dans la partie expérimentale de ces travaux, présentée dans le chapitre 7, que des valeurs par défaut peuvent être associées aux deux paramètres $ALEA\_DIV$ et NB_RUNS contrôlant la composante stochastique de la méthode. Le seul paramètre à fixer en entrée de l'algorithme par l'utilisateur est donc une borne supérieure sur le nombre de clusters recherchés, qui n'est pas nécessairement atteinte. Or, si cette borne est fixée initialement à un niveau trop bas, il est très facile de relancer après coup **Tuareg** sur un cluster résultant considéré comme pas assez homogène, et ceci d'autant plus que **Tuareg** fournit un *résultat compréhensible* sous forme de règles définies par un minimum de dimensions spécifiques pour chacun des clusters générés.

Tuareg est d'une complexité très raisonnable (cf. fin de la section 4.5). De plus, une fois les clusters créés, la *classification de nouveaux exemples est très rapide* car il suffit alors de considérer les dimensions caractéristiques des clusters, et non plus toutes les dimensions de l'espace. Notons aussi que plusieurs étapes de calculs peuvent être réalisées indépendamment les unes des autres. Les performances de **Tuareg** pourront donc être encore améliorées par la mise en œuvre d'une version parallèle.

Finalement, le tableau 4.1 résume les différentes caractéristiques de cette approche.

caractéristique	**valeur**
connaissances a priori	borne sur le nombre de clusters
présentation des résultats	ensemble de règles
complexité	$O(M \times N \log N)$
déterministe	non
incrémental	non
any-time	oui
hard	oui
prise en compte du contexte	oui
tolérance au bruit	oui
tolérance à l'effet de chaîne	oui
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	oui
tolérance aux clusters de forme quelconque	non
tolérance aux clusters concentriques	non

TAB. 4.1 – Caractéristiques associées au clustering **Tuareg**.

Les expérimentations conduites avec cette nouvelle méthode de subspace clustering sont présentées dans le chapitre 7. De par ses caractéristiques, **Tuareg** est tout à fait intéressant lorsqu'il est confronté à des bases de données pour lesquelles l'hypothèse de séparabilité des clusters sur au moins une dimension de description est vérifiée. En particulier, les résultats sont alors fournis en un temps tout à fait raisonnable, alors que d'autres méthodes de subspace clustering nécessitent un temps d'exécution beaucoup plus important pour fournir les mêmes résultats, et que des méthodes classiques de clustering ne peuvent identifier les clusters présents dans les données lorsque celles-ci sont décrites par certaines dimensions non pertinentes.

Cependant, de par ses limites, **Tuareg** obtient des résultats mitigés sur plusieurs bases de données réelles pour lesquelles l'hypothèse de séparabilité unidimensionnelle n'est pas vérifiée. C'est pourquoi avant de rentrer dans les détails de nos expérimentations, nous présentons d'abord une autre méthode de subspace clustering que nous avons mise en œuvre avec pour objectif de pallier aux limites de **Tuareg** tout en conservant ses points forts. Dans ce cadre, nous avons donc fait le choix d'une méthode capable de considérer simultanément plusieurs sous-ensembles de dimensions caractérisant les clusters, basée sur l'utilisation de modèles probabilistes, et fournissant en sortie un résultat interprétable sous forme de règles définies par un minimum d'attributs pertinents possible.

Chapitre 5

SuSE : sélection de sous-espaces en clustering statistique

Comme nous l'avons vu avec l'algorithme **Tuareg**, dans certains cas, fractionner l'ensemble des données en ne considérant leurs projections que sur un unique attribut à chaque étape n'est pas suffisant. Nous présentons donc dans ce chapitre une autre méthode de subspace clustering considérant simultanément plusieurs sous-ensembles d'attributs associés aux clusters.

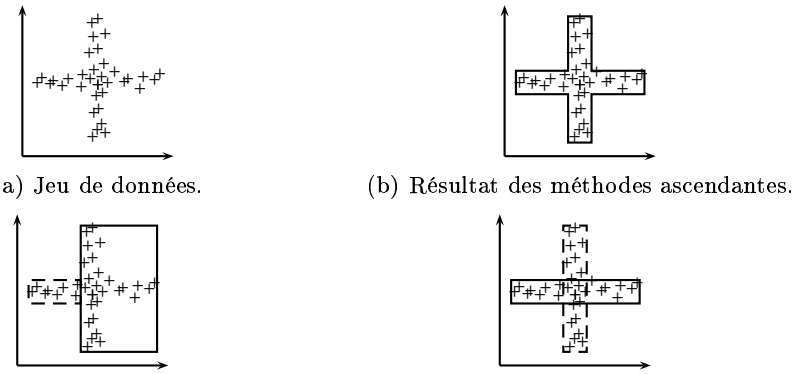
Cette nouvelle méthode est basée sur l'utilisation de modèles probabilistes, que nous avons déjà introduit dans la section 3.3 et dont nous montrons l'intérêt dans la section 5.1. Après avoir introduit les notations nécessaires pour la suite, nous présentons comment le modèle que nous utilisons est initialisé puis optimisé. La section 5.5 traite ensuite des différentes manières possibles d'effectuer de la sélection d'attributs pendant l'apprentissage, afin d'adapter la méthode au cas du subspace clustering. Nous montrons ensuite comment, parmi les différents modèles qui peuvent être envisagés pour représenter les données, celui qui correspond le mieux aux données est sélectionné. Enfin, la dernière section propose une méthode originale permettant de présenter les résultats de manière compréhensible et visuelle à l'utilisateur.

5.1 Modèles probabilistes

La méthode que nous proposons ici est basée sur l'utilisation de modèles probabilistes, qui présente plusieurs avantages importants. En particulier, nous montrons que le problème difficile de la spécification des paramètres inhérents aux méthodes de subspace clustering peut être vu comme un problème de sélection de modèle dans un cadre probabiliste. Ceci nous permet alors de proposer une méthode qui peut se passer de connaissance a priori de la part de l'utilisateur.

Par ailleurs, le fait que les méthodes probabilistes permettent naturellement le chevauchement des clusters constitue également un avantage important dans le cadre du subspace clustering. Prenons par exemple le cas présenté dans la figure 5.1, où un cluster est défini par des valeurs moyennes sur une dimension et prend des valeurs aléatoires

sur une autre dimension, et inversement pour l'autre cluster. On a donc bien dans ce cas deux clusters définis chacun dans un sous-espace (à une seule dimension) qui lui est propre, ce qui correspond tout à fait à la problématique du subspace clustering.



(a) Jeu de données. (b) Résultat des méthodes ascendantes.

(c) Résultat des méthodes descendantes. (d) Résultat des méthodes probabilistes.

FIG. 5.1 – Un cas où les méthodes existantes de subspace clustering se comportent mal contrairement aux méthodes probabilistes.

Dans un tel cas, pour les méthodes de subspace clustering ascendantes sur les dimensions, détaillées en annexes B.2, l'ensemble des objets appartient à un unique cluster parce qu'ils forment une zone continue. Ces approches tendent alors à décrire de telles données par un unique cluster défini dans un espace à deux dimensions au lieu de considérer deux clusters définis chacun dans un sous-espace différent à une seule dimension. Le problème devient alors encore plus important lorsqu'un nombre plus conséquent de dimensions sont considérées. À l'inverse, pour les méthodes de subspace clustering descendantes sur les dimensions, détaillées en annexes B.3 et pour la plupart basées sur une approche de type K-means qui ne permet pas naturellement l'intersection des clusters, les deux clusters peuvent ne pas être identifiés.

Par contre, les méthodes basées sur l'utilisation de modèles probabilistes sont capables d'identifier les deux clusters. Cependant, il est reconnu que de telles approches peuvent nécessiter de nombreuses étapes avant de converger. Par ailleurs, nous verrons l'intérêt d'adapter ces méthodes au cas du subspace clustering, par la mise en œuvre d'un modèle permettant d'identifier les sous-espaces de description spécifiques à chacun des clusters.

Enfin, le problème de la détection du bruit qui peut exister dans les données peut aussi être intégré naturellement dans un cadre probabiliste. Or nous nous attaquons également dans nos travaux à la problématique importante de fournir un résultat compréhensible des clusters identifiés, afin de permettre à l'utilisateur d'appréhender au mieux les résultats produits. Nous verrons alors dans la partie expérimentale de nos travaux, présentée dans le chapitre 7, que la détection du bruit existant dans les données permet d'obtenir des résultats plus facilement interprétables.

Notre modèle est fondé sur l'hypothèse que les données ont été générées selon des

distributions indépendantes sur chaque dimension. Le modèle est alors moins riche que le modèle classique prenant également en compte les corrélations possibles entre dimensions, présenté dans la section 3.3, mais cette hypothèse d'indépendance apporte de nombreux avantages.

Tout d'abord, nous verrons comment cela nous permet de cibler pour chaque cluster le sous-ensemble des dimensions qui lui est le plus approprié. Cette hypothèse nous donne également la possibilité de considérer différents types d'attributs caractérisant les données. Elle nous permet aussi de dériver du modèle une théorie compréhensible en représentant chaque cluster sous forme de règle (hyperrectangle dans un sous-espace de l'espace original de description des objets), puisque chaque dimension est caractérisée indépendamment des autres. Enfin, l'algorithme est alors plus rapide que l'algorithme classique car le nouveau modèle nécessite moins de paramètres ($O(M)$ au lieu du $O(M^2)$ pour M le nombre de dimensions de l'espace de description des objets) et les opérations matricielles sont évitées.

Si au contraire la méthode est confrontée à une base de données pour laquelle cette hypothèse d'indépendance des dimensions n'est pas vérifiée, alors nous verrons dans la dernière section de ce chapitre que les résultats que nous obtenons peuvent quand même apporter de l'information intéressante sur le problème considéré.

5.2 Notations

Les notations utilisées dans ce chapitre reprennent celles présentées dans la section 2.6, auxquelles nous ajoutons celles nécessaires pour représenter les modèles probabilistes associés aux clusters.

On note θ_k le modèle paramétrique associé au cluster C_k, S_k le sous-ensemble des dimensions pertinentes pour ce cluster, et R_k la règle qui lui est associée. Étant donné K le nombre total de clusters recherchés, $\theta = \{\theta_k | k \in [1, K]\}$ représente alors l'ensemble du modèle paramétrique associé à la partition $P = \{C_k | k \in [1, K]\}$.

Nous considérons ici que les données peuvent être caractérisées par deux types d'attributs : les attributs numériques et les attributs catégoriels. Sur les attributs numériques, nous faisons l'hypothèse que les données sont générées selon des distributions gaussiennes, représentées par leur centre μ et leur variance σ. Sur les attributs catégoriels, nous faisons l'hypothèse que les données sont générées selon des distributions multinomiales, représentées par un vecteur de fréquences $\overrightarrow{Freqs}$ sur l'ensemble des modalités possibles de l'attribut considéré.

Finalement, un cluster C_k est donc supposé généré par un modèle θ_k, composé d'une probabilité du cluster notée π_k, d'un centre μ_{kd} et d'une déviation standard σ_{kd} sur toute dimension numérique d, et d'une fréquence $Freqs_{kd}(mod)$ sur chaque modalité $mod \in Modalites_d$ de chaque dimension catégorielle d.

Par exemple, notons C_1 le cluster se situant à droite dans la figure 5.2. Les notations associées sont alors les suivantes :

- $D_1 = \{5, 6, 7, 8, 9\}$ représente l'ensemble des objets associés à C_1 ;
- $N_1 = 5$ dénote le nombre d'objets inclus dans le cluster ;

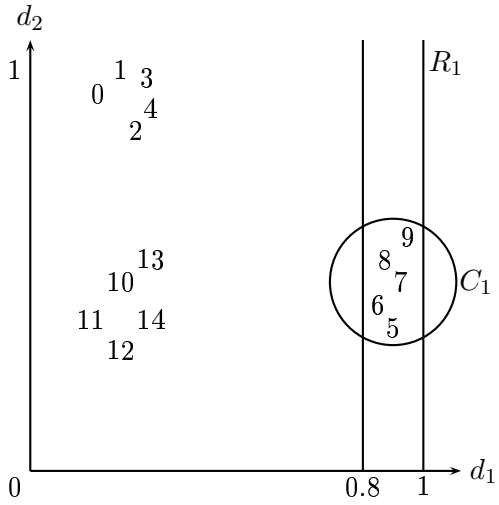


FIG. 5.2 – Notations sur l'exemple.

- $S_1 = \{d_1, d_2\}$ représente le sous-espace de description associé au cluster, puisque les objets sont proches sur ces deux dimensions ;
- $\theta_1 = (\pi_1 = 0.33, \mu_{11} = 0.9, \sigma_{11} = 0.05, \mu_{12} = 0.4, \sigma_{12} = 0.1)$ représente le modèle paramétrique associé à C_1 ;
- enfin $R_1 = \{d_1 \in [0.8, 1]\}$ est la règle qui lui est associée, définie sur une seule dimension puisque cela suffit à distinguer ce cluster des autres.

5.3 Initialisation du modèle

Dans un premier temps, nous supposons que le nombre K de clusters recherchés est connu. L'initialisation du modèle se fait aléatoirement à partir des données en utilisant une méthode de type K-means.

On sélectionne d'abord aléatoirement K objets de la base servant de centroïdes initiaux $\overrightarrow{\mu_k}$ des clusters C_k. Chaque objet $\overrightarrow{x_i}$ est ensuite associé aux membres D_k du cluster C_k dont le centroïde est le plus proche, au sens de la minimisation de la distance suivante :

$$Dist(\overrightarrow{x_i}, \overrightarrow{\mu_k}) = \sqrt{\sum_{d=1}^{M} dist(x_{id}, \mu_{kd})}$$

$$dist(x_{id}, \mu_{kd}) = \begin{cases} 0 & \text{si } d \text{ catégorielle et } x_{id} = \mu_{kd} \\ 1 & \text{si } d \text{ catégorielle et } x_{id} \neq \mu_{kd} \\ (x_{id} - \mu_{kd})^2 & \text{si } d \text{ numérique} \end{cases}$$

Les paramètres des composantes du mélange θ_k sont ensuite initialisés en fonction de l'ensemble D_k des objets associés aux clusters comme suit :

$$\pi_k = \frac{N_k}{N}$$

$$\mu_{kd} = \frac{\sum_{i \in D_k} x_{id}}{N_k} \; \forall \; d \text{ numérique}$$

$$\sigma_{kd} = \sqrt{\frac{\sum_{i \in D_k} (x_{id} - \mu_{kd})^2}{N_k}} \; \forall \; d \text{ numérique}$$

$$Freqs_{kd}(mod) = \frac{|\{i \in D_k | x_{id} = mod\}|}{N_k} \; \forall \; d \text{ catégorielle } \forall \; mod \in Modalites_d$$

Finalement, la procédure d'initialisation du modèle paramétrique associé à un ensemble de données D, et pour un nombre donné de clusters K, est présentée par l'algorithme 7.

5.4 Optimisation du modèle

L'optimisation du modèle se fait ensuite en utilisant la méthode EM [Dempster et al., 1977] présentée dans la section 3.3, mais en tenant compte de l'hypothèse d'indépendance des dimensions. Les démonstrations théoriques ayant mené aux équations présentées ci-après sont disponibles en annexes D.1. Pour rappel, la méthode EM consiste en une succession de deux phases permettant d'optimiser le modèle à chaque étape :

1. étape E (Espérance) : calculer les probabilités d'appartenance des objets aux clusters en fonction des paramètres courants du modèle θ ;
2. étape M (Maximisation) : mettre à jour les paramètres du modèle θ en fonction des nouvelles probabilités d'appartenance des objets aux clusters.

Plus formellement, l'étape d'Espérance consiste à supposer fixés les paramètres du modèle θ et à effectuer les calculs suivants pour tout objet $\overrightarrow{x_i}$ et tout cluster C_k :

$$P(\overrightarrow{x_i}|\theta_k) = \prod_{d \in S_k} P(x_{id}|\theta_{kd})$$

À ce niveau, étant donné que les clusters sont définis sur l'ensemble des dimensions de description de l'espace puisqu'aucune sélection d'attributs n'est effectuée, on a :

$$S_k = S \; \forall \; k \in [1, K]$$

$$P(x_{id}|\theta_{kd}) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma_{kd}} e^{-\frac{1}{2}\left(\frac{x_{id}-\mu_{kd}}{\sigma_{kd}}\right)^2} & \text{si } d \text{ numérique} \\ Freqs_{kd}(x_{id}) & \text{si } d \text{ catégorielle} \end{cases}$$

Algorithme 7 Initialiser

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, et K le nombre de clusters recherchés.
{Sélection aléatoire des centroïdes initiaux}
for $k = 1$ to K **do**
$\overrightarrow{\mu_k} = Random(D)$ {centroïde initial du cluster C_k sélectionné aléatoirement parmi l'ensemble des objets de D}
end for
{Association des objets aux centroïdes les plus proches}
for $i = 1$ to N **do**
$minDist = +\infty$
$c = 0$ {le cluster courant le plus proche de l'objet $\overrightarrow{x_i}$}
for $k = 1$ to K **do**
if $(Dist(\overrightarrow{x_i}, \overrightarrow{\mu_k}) < minDist)$ **then**
$minDist = Dist(\overrightarrow{x_i}, \overrightarrow{\mu_k})$
$c = k$
end if
end for
$push(D_c, \overrightarrow{x_i})$ {ajouter $\overrightarrow{x_i}$ à l'ensemble des objets D_c du cluster numéro c}
end for
{Initialisation des paramètres du modèle initial}
for $k = 1$ to K **do**
$\pi_k = \frac{N_k}{N}$
for $d = 1$ to M **do**
if (d numérique) **then**
$\mu_{kd} = \frac{\sum_{i \in D_k} x_{id}}{N_k}$
$\sigma_{kd} = \sqrt{\frac{\sum_{i \in D_k} (x_{id} - \mu_{kd})^2}{N_k}}$
else
for all $mod \in Modalites_d$ **do**
$Freqs_{kd}(mod) = \frac{|\{i \in D_k | x_{id} = mod\}|}{N_k}$
end for
end if
end for
$\theta_k = (\pi_k, \overrightarrow{\mu_k}, \overrightarrow{\sigma_k}, Freqs_k)$
end for

Sortie : modèle paramétrique initial $\theta = (\theta_1, ..., \theta_K)$.

$$P(\overrightarrow{x_i}|\theta) = \sum_{k=1}^{K} \pi_k \times P(\overrightarrow{x_i}|\theta_k)$$

$$P(\theta_k|\overrightarrow{x_i}) = \frac{\pi_k \times P(\overrightarrow{x_i}|\theta_k)}{P(\overrightarrow{x_i}|\theta)} \tag{5.1}$$

Puis l'étape de Maximisation consiste à supposer fixées les probabilités d'appartenance des objets aux clusters, et à mettre à jour les paramètres du modèle θ pour chaque cluster C_k comme suit :

$$\pi_k = \frac{1}{N} \sum_{i \in D} P(\theta_k|\overrightarrow{x_i})$$

$$\mu_{kd} = \frac{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i}) \times x_{id}}{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i})} \ \forall\ d \text{ numérique}$$

$$\sigma_{kd} = \sqrt{\frac{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i}) \times (x_{id} - \mu_{kd})^2}{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i})}} \ \forall\ d \text{ numérique}$$

$$Freqs_{kd}(mod) = \frac{\sum_{\{i \in D | x_{id} = mod\}} P(\theta_k|\overrightarrow{x_i})}{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i})} \ \forall\ d \text{ catégorielle } \forall\ mod \in Modalites_d$$

Classiquement, ces deux étapes sont itérées jusqu'à ce que la log-vraisemblance du modèle par rapport aux données $LL(\theta|D)$ s'améliore peu d'une itération à l'autre. Autrement dit, la méthode s'arrête lorsque $LL(\theta|D)^{t+1} - LL(\theta|D)^t < \delta$ pour δ strictement positif.

$$LL(\theta|D) = \sum_{i \in D} \log P(\overrightarrow{x_i}|\theta)$$

Un autre critère d'arrêt possible, de type K-means, consiste à stopper l'optimisation du modèle lorsque le cluster le plus probable de chaque objet n'est pas différent d'une itération à l'autre. Autrement dit, la méthode s'arrête lorsque les ensembles D_k d'objets associés aux clusters ne changent pas d'une itération à l'autre.

$$D_k = \{\overrightarrow{x_i} | Argmax_{j=1}^{K} P(\overrightarrow{x_i}|\theta_j) = k\}$$

Cette deuxième approche n'étant pas classique dans le cadre du clustering statistique, nous verrons dans la partie expérimentale du chapitre 7 les conséquences de son utilisation sur les résultats obtenus.

Finalement, la méthode que nous venons de décrire est résumée par l'algorithme 8 et correspond à celle qui a été mise en œuvre dans l'algorithme que nous nommons **SSC** (**S**tatistical **S**ubspace **C**lustering [Candillier et al., 2005b]) dans la suite. La valeur de δ, précisant l'écart minimal entre deux valeurs successives de la log-vraisemblance pour la poursuite de l'optimisation du modèle est fixée à 1 par défaut.

Algorithme 8 Optimiser

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, $\theta = (\theta_1, ..., \theta_K)$ un modèle paramétrique initial de ces données composé de K clusters, et δ le seuil sur les valeurs successives de la log-vraisemblance du modèle par rapport aux données.

$diff = +\infty$ {différence entre deux log-vraisemblances consécutives}

$change = 1$ {booléen exprimant si certains objets ont changé d'affectation de cluster}

while ($diff > \delta$ and $change$) **do**

$LL = LL(\theta|D)$ {log-vraisemblance du modèle courant par rapport aux données}

{Espérance}

for $i = 1$ to N **do**

for $k = 1$ to K **do**

calculer $P(\theta_k|\overrightarrow{x_i})$ selon l'équation 5.1

end for

end for

{Maximisation}

for $k = 1$ to K **do**

$\pi_k = \frac{1}{N} \sum_{i \in D} P(\theta_k|\overrightarrow{x_i})$

for all $d \in S_k$ **do**

if (d numérique) **then**

$$\mu_{kd} = \frac{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i}) \times x_{id}}{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i})}$$

$$\sigma_{kd} = \sqrt{\frac{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i}) \times (x_{id} - \mu_{kd})^2}{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i})}}$$

else

for all $mod \in Modalites_d$ **do**

$$Freqs_{kd}(mod) = \frac{\sum_{\{i \in D | x_{id} = mod\}} P(\theta_k|\overrightarrow{x_i})}{\sum_{i \in D} P(\theta_k|\overrightarrow{x_i})}$$

end for

end if

end for

end for

{Critères d'arrêt}

$diff = LL(\theta|D) - LL$

$change = 0$

for $k = 1$ to K **do**

for all $i \in D_k$ **do**

$maxP = P(\overrightarrow{x_i}|\theta_k)$

for $c = 1$ to K **do**

if ($c \neq k$ and $P(\overrightarrow{x_i}, \theta_c) > maxP$) **then**

$maxP = P(\overrightarrow{x_i}, \theta_c)$

$move(\overrightarrow{x_i}, D_k, D_c)$ {réaffecter $\overrightarrow{x_i}$ de D_k à D_c}

$change = 1$

end if

end for

end for

end for

end while

Sortie : modèle paramétrique optimisé $\theta = (\theta_1, ..., \theta_K)$.

5.5 Sélection d'attributs

En utilisant un tel modèle, toutes les dimensions n'ont pas la même importance dans la détermination de l'appartenance des objets aux clusters. En effet, le modèle pondère de manière intrinsèque l'importance de chaque dimension pour chaque cluster. Par exemple pour une dimension numérique, une déviation standard élevée va induire un poids faible de la dimension alors qu'une faible déviation standard va induire un poids important de la dimension pour la caractérisation de l'appartenance des objets au cluster.

Nous pouvons cependant aller plus loin et sélectionner un sous-ensemble des dimensions les plus pertinentes pour chaque cluster.

Une solution pour cela consiste à comparer sur chaque dimension la vraisemblance de notre modèle (gaussien ou multinomial) par rapport à celle d'un modèle uniforme. Ainsi, si la vraisemblance d'un modèle uniforme est plus importante que celle de notre modèle, alors la dimension est considérée comme non pertinente pour le cluster.

La vraisemblance d'un modèle θ sur un cluster C_k et une dimension d est calculée comme suit :

$$LL(\theta|C_k, d) = \sum_{i \in D_k} \log P(x_{id}|\theta)$$

1. Dans le cas d'un modèle uniforme sur une dimension numérique θ_{U_n}, comme nous avons fait l'hypothèse que les valeurs des objets sur les dimensions numériques sont normalisées entre 0 et 1, on a :

   $$P(x_{id}|\theta_{U_n}) = 1 \ \forall \ i \in D \ \forall \ d \text{ numérique}$$

   Par conséquent, $LL(\theta_{U_n}|C_k, d) = 0$, et donc une dimension numérique d est considérée comme pertinente pour un cluster C_k si :

   $$LL(\theta_{kd}|C_k, d) > 0$$

2. Une distribution uniforme θ_{U_c} sur une dimension catégorielle d est définie comme suit :

   $$P(x_{id}|\theta_{U_c}) = \frac{1}{|Categories_d|}$$

   On a donc :

   $$LL(\theta_{U_c}|C_k, d) = -N_k \times \log |Categories_d|$$

   Pour notre modèle multinomial sur les dimensions catégorielles, on a :

   $$LL(\theta_{kd}|C_k, d) = \sum_{i \in D_k} \log Freks_{kd}(x_{id})$$

   Comme $LL(\theta_{kd}|C_k, d)$ est toujours supérieur à $LL(\theta_{U_c}|C_k, d)$ et que les deux valeurs sont négatives, il est possible d'introduire une constante $0 < \alpha < 1$ et de poser que la dimension catégorielle d est pertinente pour le cluster C_k si :

   $$LL(\theta_{kd}|C_k, d) > \alpha \times LL(\theta_{U_c}|C_k, d)$$

Cependant, outre la difficulté de fixer ce paramètre α pour les dimensions catégorielles, un inconvénient de cette solution pour sélectionner les dimensions pertinentes des clusters est qu'elle nécessite de parcourir l'ensemble des valeurs des objets, ce qui induit un temps de calcul supplémentaire important.

Nous proposons donc une autre alternative qui effectue la sélection des dimensions pertinentes des clusters directement en fonction des paramètres courants du modèle. Pour cela, nous ajoutons au modèle une matrice de poids W de l'ensemble des dimensions sur l'ensemble des clusters. Ces poids W_{kd} correspondent au rapport entre déviation standard locale et globale par rapport à μ_{kd} sur les dimensions numériques, et à la fréquence relative de la catégorie la plus probable du cluster sur les dimensions catégorielles :

$$W_{kd} = \begin{cases} 1 - \frac{\sigma_{kd}^2}{\Sigma_{kd}^2} \text{ , avec } \Sigma_{kd}^2 = \frac{1}{N}\sum_{i \in D}(x_{id} - \mu_{kd})^2 & \text{si } d \text{ numérique} \\ \frac{Freqs_{kd}(mod) - Frequences_d(mod)}{1 - Frequences_d(mod)} & \text{si } d \text{ catégorielle} \\ \text{avec } mod = Argmax_{\{m \in Modalites_d\}} Freqs_{kd}(m) & \end{cases} \tag{5.2}$$

Enfin, nous ajoutons au modèle le paramètre R qui indique combien de dimensions pertinentes considérer pour chaque cluster. L'ensemble S_k des dimensions associées au cluster C_k contient donc les R dimensions qui maximisent leur poids W_{kd}.

Par ce modèle, nous posons que tous les clusters ont le même nombre de dimensions pertinentes, même si les dimensions sélectionnées peuvent être différentes d'un cluster à l'autre. Si ce n'est pas le cas, alors quelques dimensions non pertinentes peuvent être sélectionnées pour certains clusters. Cependant, l'influence de telles dimensions non pertinentes sera moins importante que celle des dimensions pertinentes puisque, comme nous l'avons expliqué précédemment, l'importance des dimensions pour chaque cluster est intrinsèquement déterminée par la forme de la distribution des membres des clusters sur les dimensions.

Cette sélection des attributs pertinents des clusters peut être effectuée uniquement dans la phase d'initialisation du modèle, puis ces attributs sélectionnés sont conservés tout au long de l'optimisation du modèle, ou bien au contraire, ces sous-ensembles d'attributs pertinents associés aux clusters peuvent être mis à jour à chaque itération de la méthode EM. C'est le premier choix que nous avons retenu, les performances de la méthode s'en trouvant peu changées alors que le temps d'exécution de la méthode est nettement amélioré.

Dans la suite, nous nommons **SuSE** (**Su**bspace **S**election embedded in an **E**M algorithm [Candillier et al., 2006b]) la méthode de clustering mettant en œuvre cette sélection d'attributs. Au niveau algorithmique, seule une étape est ajoutée lors de l'initialisation du modèle, afin d'effectuer la sélection des dimensions les plus pertinentes des clusters. La nouvelle procédure d'initialisation est fournie par l'algorithme 9. Ensuite, pour l'optimisation du modèle, les étapes sont les mêmes, mais cette fois, les ensembles S_k de dimensions associées aux clusters sont spécifiques à chacun des clusters, et les

calculs effectués lors des étapes d'Espérance et de Maximisation s'en trouvent donc modifiés.

Algorithme 9 Initialiser2

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, K le nombre de clusters recherchés, et R le nombre de dimensions pertinentes associées aux clusters.

```
Initialiser(D,K)
for k = 1 to K do
  for d = 1 to M do
    calculer W_kd selon l'équation 5.2
  end for
  sélectionner les R dimensions de poids maximum et les stocker dans S_k
  θ_k = (π_k, μ_k, σ_k, Freqs_k, W_k, S_k)
end for
```

Sortie : modèle paramétrique initial $\theta = (\theta_1, ..., \theta_K)$.

5.6 Sélection de modèle

Les résultats d'une exécution de la méthode (initialisation puis optimisation du modèle) dépendent fortement de la solution initiale proposée, c'est-à-dire que le résultat peut être un optimum local. Pour tenter d'atteindre l'optimum global, il est donc nécessaire d'itérer la méthode un certain nombre de fois et de sélectionner le résultat le plus approprié. Le processus d'optimisation du modèle est donc exécuté plusieurs fois (NB_ITER) à partir de différentes initialisations aléatoires, puis le modèle θ qui optimise sa log-vraisemblance $LL(\theta|D)$ par rapport aux données D est conservé.

Autrement dit, la méthode consiste à générer plusieurs modèles possibles pour représenter les données, puis à choisir le modèle le plus vraisemblable. Dans un tel cas, comme il s'agit de comparer différents modèles ayant la même complexité (c'est-à-dire le même nombre de paramètres), utiliser le critère de vraisemblance (qui constitue un critère de la qualité interne du clustering) est suffisant pour déterminer le modèle le plus approprié aux données.

Si au contraire on souhaite évaluer le modèle le plus approprié aux données parmi un ensemble de modèles de complexités différentes, alors la simple utilisation de la vraisemblance du modèle par rapport aux données n'est pas suffisante. Il faut dans ce cas trouver un compromis entre la vraisemblance du modèle par rapport aux données et sa complexité. On retrouve donc à ce niveau le compromis classique entre spécificité et généralisation du modèle proposé.

Plusieurs critères ont été proposés pour cela qui ajoutent au calcul de la vraisemblance du modèle par rapport aux données un terme qui tend à pénaliser les modèles les plus complexes. Les différents critères se différencient donc sur leur façon de pénaliser les modèles les plus complexes. Parmi ceux-là, on peut par exemple citer le critère AIC (Akaike Information Criterion) [Akaike, 1970], ou bien le critère ICL (Integrated Completed Likelihood) [Biernacki et al., 2000], mais le critère le plus connu et le plus utilisé

dans ce cadre est sans doute le critère BIC (Bayesian Information Criterion) [Schwartz, 1979], défini comme suit :

$$BIC(\theta|D) = -2 \times LL(\theta|D) + M_\theta \times \log N \tag{5.3}$$

La valeur de ce critère doit êre minimisée pour optimiser la pertinence du modèle par rapport aux données. M_θ représente le nombre de paramètres indépendants du modèle, et est calculé comme suit dans notre cas :

$$M_\theta = \sum_{k=1}^{K} \sum_{d \in S_k} \begin{cases} 2 & \text{si } d \text{ numérique} \\ |Modalites_d| & \text{si } d \text{ catégorielle} \end{cases}$$

Dans le cas des deux méthodes **SSC** et **SuSE** que nous proposons, ce critère BIC peut donc être utilisé pour déterminer automatiquement les paramètres du modèle les plus appropriés aux données. Les deux méthodes nécessitent de spécifier le nombre K de clusters recherchés, et la deuxième méthode **SuSE** nécessite également la spécification du paramètre R qui indique le nombre de dimensions à sélectionner pour chaque cluster.

Un avantage important de l'utilisation des modèles probabilistes par rapport aux autres méthodes existantes de subspace clustering est alors que la détermination des valeurs les plus appropriées pour ces paramètres peut être vu comme un problème de sélection de modèle, ce qui nous permet de proposer une méthode ne nécessitant aucune connaissance a priori de la part de l'utilisateur.

Pour trouver le modèle qui correspond le mieux aux données, on considère donc plusieurs modèles avec différentes valeurs pour les paramètres K et R, puis le modèle minimisant la valeur BIC est finalement conservé. Ces différentes générations de modèles sont indépendantes les unes des autres, ce qui justifie la mise en œuvre d'exécutions parallèles permettant d'améliorer les performances de la méthode.

Étant donnés NB_ITER le nombre d'itérations d'une étape de la méthode (initialisation et optimisation de modèle), et Max_K le nombre maximum de clusters recherchés, l'algorithme **SSC** est finalement présenté par l'algorithme 10. Et étant donné également Max_R le nombre maximum de dimensions pertinentes associées aux clusters, l'algorithme **SuSE** est lui présenté par l'algorithme 11. Le nombre d'itérations NB_ITER de l'optimisation du modèle avec initialisation aléatoire est fixé à 10 par défaut.

5.7 Présentation des résultats

Nous proposons ensuite une méthode permettant de représenter les clusters C_k à l'utilisateur sous forme de règles R_k définies par un minimum de dimensions parmi les plus pertinentes. La construction de ces règles R_k se fait à partir de l'ensemble des objets D_k associés aux clusters.

Une phase préliminaire de détection du bruit existant dans les données est donc nécessaire afin d'éviter de former ces règles associées aux clusters en prenant en compte

Algorithme 10 SSC

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, δ le seuil sur les valeurs successives de la log-vraisemblance du modèle par rapport aux données, NB_ITER le nombre d'itérations d'une étape de la méthode, et Max_k le nombre maximum de clusters recherchés.

$minBIC = +\infty$ {valeur minimale courante du critère BIC}
$\theta_o = \emptyset$ {modèle courant optimal}
for $K = 1$ to Max_K **do**
  for $i = 1$ to NB_ITER **do**
    θ = Initialiser(D,K)
    θ = Optimiser(D,θ,δ)
    calculer $BIC(\theta|D)$ selon l'équation 5.3
    if $(BIC(\theta|D) < minBIC)$ **then**
      $minBIC = BIC(\theta|D)$
      $\theta_o = \theta$
    end if
  end for
end for

Sortie : modèle paramétrique optimal associé aux données θ_o.

Algorithme 11 SuSE

Entrée : $D = (\overrightarrow{x_1}, ..., \overrightarrow{x_N})$ un ensemble d'objets, δ le seuil sur les valeurs successives de la log-vraisemblance du modèle par rapport aux données, NB_ITER le nombre d'itérations d'une étape de la méthode, Max_k le nombre maximum de clusters recherchés, et Max_R le nombre maximum de dimensions pertinentes associées aux clusters.

$minBIC = +\infty$ {valeur minimale courante du critère BIC}
$\theta_o = \emptyset$ {modèle courant optimal}
for $K = 1$ to Max_K **do**
  for $R = 1$ to Max_R **do**
    for $i = 1$ to NB_ITER **do**
      θ = Initialiser2(D,K,R)
      θ = Optimiser(D,θ,δ)
      calculer $BIC(\theta|D)$ selon l'équation 5.3
      if $(BIC(\theta|D) < minBIC)$ **then**
        $minBIC = BIC(\theta|D)$
        $\theta_o = \theta$
      end if
    end for
  end for
end for

Sortie : modèle paramétrique optimal associé aux données θ_o.

des objets non pertinents pour ces clusters, dont la présence rendrait les résultats moins compréhensibles.

Pour ce faire, on utilise classiquement l'hypothèse que le bruit est généré de façon uniforme sur l'espace de description des objets. Pour déterminer si un objet constitue du bruit, la méthode consiste donc à comparer sa probabilité d'appartenance à chacun des clusters formés avec sa probabilité d'appartenance à un cluster qui serait défini par une distribution uniforme sur l'ensemble des dimensions de l'espace. Par conséquent, un objet $\overrightarrow{x_i}$ est considéré comme étant du bruit si :

$$P(\overrightarrow{x_i}|\theta_k) < 1 \ \forall \ k \in [1, K]$$

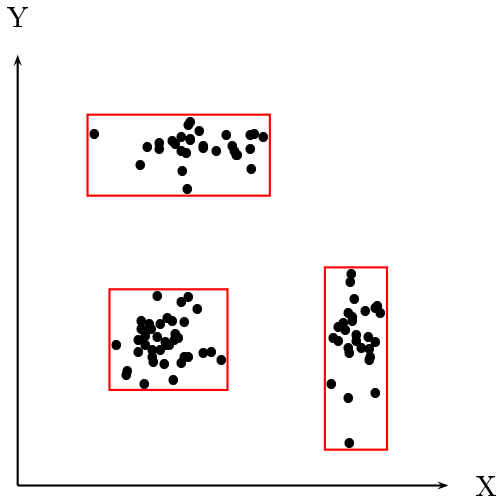
Les objets considérés comme étant du bruit sont alors supprimés des ensembles d'objets D_k associés aux clusters. Nous verrons dans la partie expérimentale de nos travaux présentée dans le chapitre 7 l'intérêt d'une telle détection du bruit pour la présentation d'un résultat aussi pertinent que possible.

Ensuite, les règles initiales sont formées à partir de ces nouveaux ensembles d'objets D_k : sur une dimension numérique, on associe à la règle l'intervalle minimal contenant l'ensemble des valeurs des objets de D_k sur la dimension ; et sur une dimension catégorielle, nous avons choisi d'associer à la règle la modalité la plus fréquente parmi l'ensemble des valeurs des objets de D_k sur la dimension ; ce choix est discutable : nous pourrions associer à la règle l'union des modalités présentes parmi les membres du cluster, ou bien l'union des modalités ayant une fréquence minimale à l'intérieur du cluster, mais ainsi, le concept présenté sur une dimension catégorielle est tout à fait porteur de sens pour l'utilisateur : il met en avant la catégorie majoritaire dans le cluster sur la dimension catégorielle considérée.

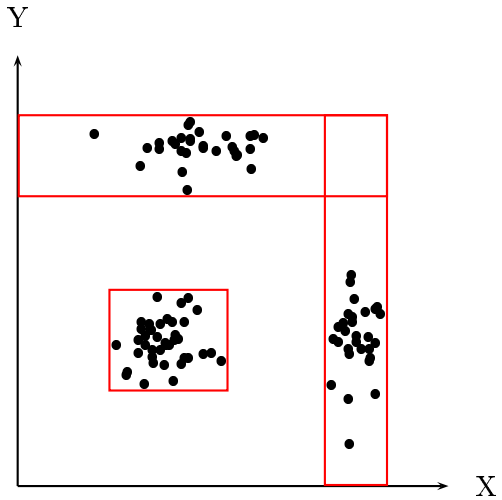
Nous proposons d'ajouter ensuite une phase de simplification de ces règles, afin qu'elles soient décrites par un minimum d'attributs parmi les plus pertinents. L'esprit est alors le même que dans la phase de description minimale de **Tuareg**, sauf qu'ici, il s'agit d'un post-traitement, alors que dans **Tuareg**, cette phase était menée pendant l'apprentissage. Pour ce faire, on calcule d'abord le support courant de la règle (l'ensemble des objets inclus dans la règle peut être plus large que D_k). Puis pour chaque cluster C_k, on supprime, dans l'ordre croissant de leur poids W_{kd}, les dimensions d de la règle si leur suppression ne modifie pas son support. Ainsi, un minimum des dimensions parmi les plus pertinentes de chaque cluster est conservé. La figure 5.3 présente le résultat d'une telle simplification sur un exemple, qui s'avère alors identique à celui obtenu par **Tuareg**.

Enfin, cette description minimale des clusters nous donne également la possibilité de proposer une méthode de visualisation graphique des résultats qui ne soit pas trop coûteuse. L'idée est de proposer à l'utilisateur une visualisation en deux ou trois dimensions des règles représentant les clusters ciblés, en présentant d'abord les projections sur lesquelles les différences entre les clusters apparaissent visuellement, et sur lesquelles les spécificités des différents clusters apparaissent également.

Pour ce faire, nous projetons donc les règles sur l'ensemble des couples de dimensions utilisées dans la définition des règles associées aux clusters (ensemble qui est souvent fortement réduit lors de la phase précédente de simplification des règles), puis nous



(a) Résultat avant simplification.



(b) Résultat après simplification.

FIG. 5.3 – Exemple de simplification des règles.

calculons pour chaque couple le taux t_1 d'intersection entre les règles projetées, et le taux t_2 de zone couverte par aucune règle projetée. Les couples de dimensions peuvent ensuite être ordonnés par ordre décroissant des valeurs $(1-t_1)\times t_2$. Les couples de dimensions de poids maximum sont alors ceux qui constituent un bon compromis entre la séparabilité (critère t_1) et la spécificité (critère t_2) des clusters projetés sur ces dimensions.

5.8 Bilan

Dans ce chapitre, nous avons mis en avant l'intérêt d'utiliser des modèles probabilistes dans le cadre du subspace clustering. En particulier, nous avons vu que le problème difficile de la spécification des paramètres inhérents aux méthodes de subspace clustering peut être vu comme un problème de sélection de modèle dans un cadre probabiliste.

Les paramètres nécessaires à notre méthode sont deux bornes supérieures : Max_K représente le nombre maximum de clusters présents dans les données, et Max_R représente le nombre maximum de dimensions pertinentes associées aux clusters. La méthode consiste alors à générer un ensemble de modèles dont le paramètre K du nombre de clusters recherchés varie entre 1 et Max_K, et le paramètre R du nombre de dimensions pertinentes des clusters varie entre 1 et Max_R. Le modèle le plus approprié aux données est alors sélectionné en utilisant le critère statistique BIC.

Si la méthode est confrontée à un jeu de données de petite taille, ou si l'utilisateur n'a pas de délai imposé pour l'exécution de la méthode, alors on peut poser $Max_K = N$ et $Max_R = M$, et ainsi tester l'ensemble des modèles possibles pour représenter les données. Sinon, une perspective de recherche intéressante de ces travaux serait de proposer une méthode efficace permettant d'atteindre la valeur optimum du critère BIC, identifiant le nombre de clusters et le nombre de dimensions pertinentes des clusters les plus appropriés, sans avoir à tester tous les modèles possibles pour ces deux paramètres.

Nous avons également montré l'intérêt de supposer que les données sont générées de façon indépendante sur chacune des dimensions. Tout d'abord c'est sous cette hypothèse que nous avons proposé notre nouvelle méthode de sélection des attributs pertinents des clusters. Ensuite cela nous permet d'accélérer la méthode et de considérer plusieurs types d'attributs différents décrivant les données. Enfin, cette hypothèse nous permet de cibler des clusters qui pourront ensuite être naturellement représentés sous forme de règles, permettant ainsi la présentation des résultats sous forme compréhensible et visuelle à l'utilisateur.

Dans le cas où des corrélations entre dimensions existent, deux cas peuvent se présenter et sont illustrés par la figure 5.4. Sur le premier exemple, la corrélation existant entre les deux dimensions ne perturbe pas l'identification des clusters car ceux-ci sont suffisamment séparés pour être détectés en utilisant notre modèle qui ne prend pas en compte de telles corrélations. Sur le second exemple par contre, deux clusters sont trop proches pour que cette approximation du modèle soit suffisante, et ils ne sont donc pas retrouvés. Cependant, nous pensons que la technique de visualisation graphique des résultats que nous proposons peut s'avérer tout à fait intéressante dans un tel cas car elle permet à l'utilisateur de repérer visuellement une telle corrélation.

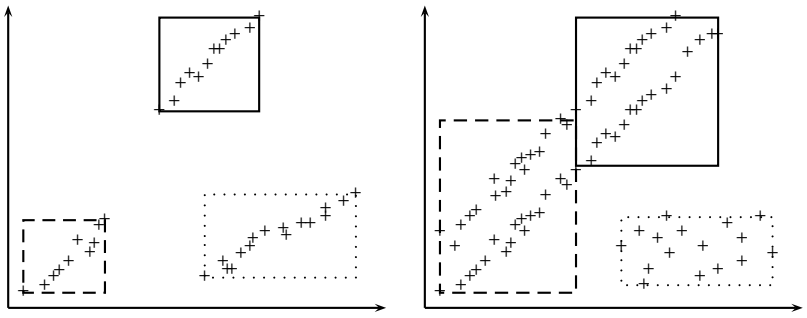


FIG. 5.4 – Résultats de **SuSE** lorsqu'une corrélation entre dimensions existe.

Contrairement à **Tuareg**, **SuSE** est capable de détecter les clusters présents dans les exemples de la figure 5.5. Dans le premier cas les clusters sont identifiés. Le deuxième cas vient d'être discuté : les clusters pourront être identifiés s'ils ne sont pas trop proches l'un de l'autre. Et dans le troisième cas, **SuSE** retournera trois clusters, celui en L étant divisé en deux, résultat qui nous semble tout à fait raisonnable dans le cadre que nous nous sommes fixé d'extraction de connaissances.

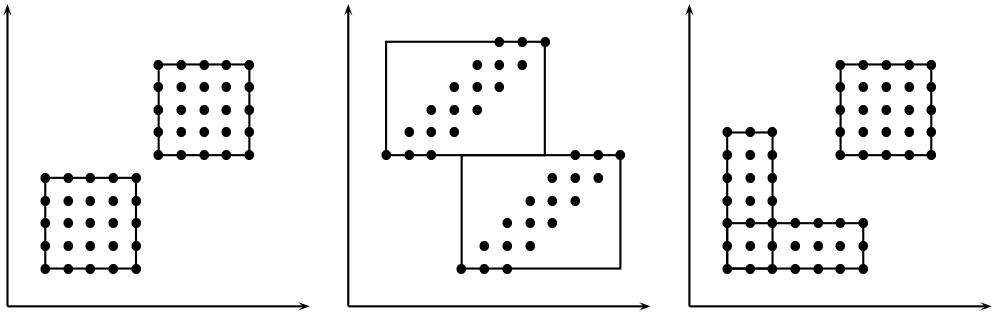


FIG. 5.5 – Exemples de résultats de **SuSE**.

Finalement, le tableau 5.1 résume les différentes caractéristiques de la nouvelle méthode de subspace clustering **SuSE** que nous venons de présenter. Pour une valeur donnée de K et de R, la complexité de la phase d'initialisation de la méthode est en $O(N \times M \times K)$ et celle de la phase d'optimisation du modèle en $O(N \times R \times K)$. Les valeurs de K variant entre 1 et Max_K et les valeurs de R variant entre 1 et Max_R, la complexité globale de la méthode est donc en $O(N \times (M + Max_R^2) \times Max_K^2)$.

Avant de présenter dans le chapitre 7 les expérimentations que nous avons conduites avec ces deux nouvelles méthodes de subspace clustering **Tuareg** et **SuSE** que nous proposons, nous allons aborder dans le chapitre 6 différentes façons possibles d'étendre de telles méthodes dans différents cas : celui d'un apprentissage autre que non supervisé, puis face à des données présentées autrement que sous forme d'attributs-valeurs.

Les principales questions abordées ensuite dans la partie expérimentale de nos tra-

caractéristique	**valeur**
connaissances a priori	bornes sur le nombre de clusters Max_K et de dimensions pertinentes des clusters Max_R
présentation des résultats	ensemble de règles
complexité	$O(N \times (M + Max_R^2) \times Max_K^2)$
déterministe	non
incrémental	oui
any-time	oui
hard	non
prise en compte du contexte	oui
tolérance au bruit	oui
tolérance à l'effet de chaîne	oui
tolérance aux clusters de tailles variées	oui
tolérance aux clusters de densités variées	oui
tolérance aux clusters de forme quelconque	non
tolérance aux clusters concentriques	non

TAB. 5.1 – Caractéristiques associées au clustering **SuSE**.

vaux sont les suivantes :

1. quelle est l'influence des paramètres de **Tuareg** et **SuSE** ?
2. comment se comportent ces méthodes selon le nombre d'objets, le nombre de dimensions, le nombre et les caractéristiques des clusters présents dans les jeux de données ?
3. sont-elles robustes à l'ajout de dimensions non pertinentes, à la présence de bruit dans les données, ou de valeurs manquantes ?
4. en quoi sont-elles intéressantes comparées aux méthodes existantes de clustering et de subspace clustering ?
5. sont-elles capables d'extraire de la connaissance à partir de bases de données réelles et de la restituer de manière compréhensible ?

Chapitre 6

Extensions des méthodes

Dans ce chapitre, nous abordons deux types d'extensions qui peuvent être apportées aux méthodes de clustering. La première porte sur l'application de telles méthodes dans le cadre d'apprentissages supervisés, semi-supervisés ou partiellement supervisés. La seconde porte sur leur adaptation possible dans le cadre de données représentées autrement que sous forme d'attributs-valeurs. Plus particulièrement, nous nous sommes intéressé au cas des données semi-structurées dans le cadre de la classification de documents XML à partir de leur structure.

6.1 Extensions au supervisé

Dans le cas où toute ou partie de l'information d'appartenance des objets à des classes prédéterminées est connue, il est possible d'adapter les méthodes de clustering pour prendre en compte de telles informations.

Dans un cas semi-supervisé, si nous faisons l'hypothèse qu'une classe prédéterminée peut être approximée par une unique distribution, c'est-à-dire si une classe correspond à un unique cluster, alors l'extension des méthodes est directe : les objets partageant la même classe sont associés au même cluster, et l'apprentissage est alors concentré sur l'ensemble des objets pour lesquels l'information d'appartenance aux classes prédéterminées n'est pas connue.

Si la méthode de clustering utilisée est statistique, comme pour **SuSE**, alors elle peut être étendue en prenant en compte des probabilités d'appartenance des objets aux classes prédéterminées, et même si ces informations ne sont que partielles. Dans ce cas, il s'agit de figer les probabilités d'appartenance des objets aux clusters qui sont connues (ou partiellement connues), c'est-à-dire de ne jamais les modifier lors de l'étape d'Estimation de l'algorithme EM.

Dans un cas supervisé, les résultats peuvent être beaucoup plus sensibles à l'hypothèse d'unicité des classes. En effet, dans de nombreux cas, approximer une classe par une unique distribution n'est pas suffisant. Il est souvent plus approprié d'utiliser un mélange de distributions pour représenter les classes, car ceci permet alors de cibler les concepts disjonctifs qui peuvent être présents dans les données. Deux méthodes sont

alors envisageables pour effectuer un apprentissage supervisé avec une méthode donnée de clustering.

La première approche d'apprentissage consiste simplement à exécuter le clustering sur chaque classe indépendamment [Hastie and Tibshirani, 1996]. Mais dans ce cas, les modèles correspondant à chaque classe ne sont pas construits en les opposant à ceux des autres classes, et les concepts ciblés risquent de se recouvrir. Une autre alternative consiste alors à lancer le clustering directement sur l'ensemble de la base en traitant l'attribut de classe comme un autre attribut, éventuellement en lui donnant plus de poids en le répliquant. Dans ce deuxième cas, les clusters ne sont alors pas associés directement aux classes, mais à chaque cluster est associée une probabilité d'appartenance à chacune des classes.

Ensuite, au niveau de la classification de nouveaux exemples, si la méthode de clustering utilisée est statistique, comme pour **SuSE**, alors dans le premier cas, il s'agit de calculer la probabilité d'appartenance de chaque nouvel objet à chaque mélange de distributions associé à chaque classe, et l'objet est ensuite associé à la classe ayant mené à la probabilité d'appartenance la plus forte. Dans le second cas, où à chaque cluster est associée une probabilité d'appartenance à chacune des classes, les probabilités d'appartenance des objets aux classes sont calculées comme suit :

$$P(c|\overrightarrow{x_i}) = \sum_{k=1}^{K} P(\overrightarrow{x_i}|\theta_k) \times P(c|\theta_k)$$

Enfin, si la méthode de clustering utilisée n'est pas statistique, les probabilités d'appartenance des objets aux clusters sont généralement remplacées par des distances, un objet étant alors associé au cluster le plus proche selon cette fonction de distance.

6.2 Extensions aux données semi-structurées

Dans le cadre du projet INEX (*INitiative for the Evaluation of XML retrieval*), une problématique nouvelle a émergé concernant la classification de documents semi-structurés en se basant sur leur contenu ou leur structure [Denoyer et al., 2006].

Nous présentons dans cette section nos travaux dans le cadre de la classification supervisée et non supervisée de documents XML à partir de leur structure uniquement, et en utilisant des modèles probabilistes [Candillier et al., 2005a]. L'entrée de l'algorithme correspond donc à une collection de documents XML qu'il s'agit de séparer en différents groupes selon la proximité de leur structure arborescente.

6.2.1 Représentation des documents XML

La figure 6.1 présente un exemple de document XML représenté sous forme arborescente. La plupart des méthodes existantes qui s'attaquent à la problématique de la classification de documents XML travaillent directement sur les arbres représentant ces documents. Certaines méthodes se basent sur l'utilisation de distances pour comparer la proximité entre les arbres représentant les documents : la distance d'édition, nombre

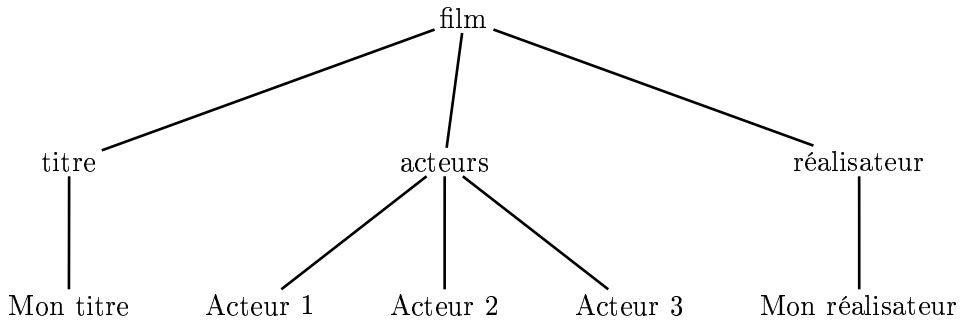


FIG. 6.1 – Exemple de document XML représenté sous forme arborescente.

minimum de mutations nécessaires pour changer un arbre en un autre [Nierman and Jagadish, 2002, Dalamagas et al., 2004], ou bien le nombre de chemins en commun entre deux arbres [Flesca et al., 2002, Lian et al., 2004, Costa et al., 2004]. D'autres méthodes se donnent pour objectif de découvrir des sous-arbres fréquents dans l'ensemble des données [Termier et al., 2002, Zaki and Aggarwal, 2003].

Nous avons étudié l'utilisation de différents types de représentations pour manipuler des documents XML. Notre proposition consiste à transformer les arbres en ensembles d'attributs-valeurs. En effectuant de telles transformations, nous perdons évidemment certaines propriétés contenues dans les documents XML initiaux, mais nous pensons qu'avec un codage pertinent, les informations principales contenues dans les arbres peuvent être conservées, et le problème simplifié. De plus, nous pouvons alors bénéficier des atouts des méthodes de classification existantes capables de traiter des données présentées sous forme d'attributs-valeurs.

Dans [Doucet and Ahonen-Myka, 2002], les auteurs ont utilisé une telle idée pour prendre en compte la structure des documents XML lors de leur classification, mais la représentation qu'ils ont choisi pour les arbres est un simple *sac de labels*. Dans [Gilleron et al., 2006], au contraire, la représentation en attributs-valeurs utilisée pour transformer les arbres est plus riche, et permet d'obtenir de très bons résultats dans un cadre d'extraction d'informations.

La représentation des arbres que nous proposons d'utiliser est alors la suivante : nous associons à chacun des arbres XML rencontrés :

- l'ensemble des labels présents dans l'arbre,
- mais aussi l'ensemble des relations père-fils et des relations frères-suivant (*next-sibling*), dont les domaines de définition sont l'ensemble des paires de labels associés aux nœuds,
- ainsi que l'ensemble des chemins (et sous-chemins) démarrant de la racine, dont le domaine de définition correspond à l'ensemble des séquences finies de labels associés aux nœuds.

Nous créons donc autant de nouveaux attributs que de relations distinctes entre labels sont rencontrées dans l'ensemble des données XML considérées. Les valeurs de ces nouveaux attributs dans les différents documents correspondent alors au nombre de

leur occurence dans ces documents. Enfin, nous définissons également autant de nouveaux attributs qu'il existe de positions absolues différentes des nœuds dans les arbres. L'identifiant d'un tel nœud peut par exemple être codé par une séquence d'entiers : la racine est codée 0, son premier fils 0.0, son second fils 0.1, etc. Pour chaque identifiant d'une position de nœud, la valeur de l'attribut pour un document correspond à l'arité du nœud, c'est-à-dire au nombre de ses fils dans le document. Les nouveaux attributs créés prennent donc finalement tous leurs valeurs dans l'ensemble des nombres naturels. Intuitivement, une telle représentation devrait permettre de différencier par exemple :

- deux ensembles de documents utilisant des labels différents, ou pour lesquels le nombre de certains labels sont différents ;
- un ensemble de documents pour lequel une relation donnée (père-fils ou frère-suivant) entre certains labels est permise d'un autre ensemble pour lequel cette relation n'est pas permise ;
- ou bien un ensemble de documents pour lequel le nombre de fils d'un nœud donné est différent de celui d'un autre ensemble de documents.

L'utilisation d'une telle représentation nous permet alors d'appliquer différentes méthodes existantes capables de traiter des données présentées sous forme d'attributs-valeurs pour classifier des ensembles de documents XML. Nous pouvons ainsi bénéficier des atouts de ces méthodes. Cependant, comme de nombreux attributs sont formés par une telle technique de transformation des arbres en attributs-valeurs, les méthodes de classification utilisées ensuite doivent être capables de faire face à de nombreux attributs, et faire de la sélection d'attributs pendant l'apprentissage. Dans le cadre d'un apprentissage supervisé, C4.5 [Quinlan, 1993] ou C5 boosté [Quinlan, 2004] sont alors tout à fait appropriés, de même qu'un algorithme de subspace clustering comme **SuSE**, présenté dans le chapitre précédent, dans un cadre non supervisé.

6.2.2 Clustering de documents XML

Par ailleurs, nous pouvons ici profiter des connaissances supplémentaires que nous possédons du domaine étudié pour adapter **SuSE** dans ce cadre. En effet, au lieu de considérer collectivement l'ensemble important des nouveaux attributs créés pour représenter les documents XML, nous pouvons considérer les différents ensembles d'attributs indépendamment pendant l'apprentissage. Outre l'accélération de la méthode et l'amélioration de l'interprétation des résultats, cela nous permet d'éviter de considérer l'ensemble des attributs simultanément qui peut s'avérer très redondant.

D'autre part, nous sommes confrontés dans ce cadre à des jeux de données très importants car caractérisés par de nombreux documents ainsi que de nombreux nouveaux attributs, qui peuvent être séparés en de nombreux groupes différents. Or face à ce type de jeux de données, les méthodes de clustering statistique peuvent nécessiter de nombreuses étapes avant de converger. Afin d'accélérer la méthode tout en conservant des résultats interprétables, nous proposons de combiner **SuSE** avec une méthode hiérarchique descendante. **SuSE** est alors utilisé récursivement pour diviser les clusters déjà formés en deux.

Finalement, notre proposition consiste donc à adapter **SuSE** de telle façon que le jeu

de données soit divisé en deux de façon récursive et selon un ensemble donné d'attributs associé aux arbres. Ainsi, à chaque étape, seule une partie des attributs, une partie de plus en plus réduite des objets, et une partition en deux de ces objets sur ces attributs sont considérés, et la sortie de la méthode est alors un arbre de décision dans lequel chaque nœud représente un test sur un ensemble restreint d'attributs.

Avant de donner plus de détails sur la procédure de clustering proposée, introduisons quelques notations. A dénote l'ensemble des attributs possibles associés aux documents XML. Comme nous l'avons introduit, A peut être partitionné en différents groupes de complexité croissante :

- nous appelons A_1 l'ensemble des labels présents dans les documents XML,
- A_2 est l'ensemble des relations père-fils présents dans les arbres,
- A_3 est l'ensemble des relations frères-suivant,
- A_4 est l'ensemble des positions de nœuds,
- et A_5 est l'ensemble des chemins démarrant de la racine.

A est donc composé de $SA = 5$ ensembles d'attributs. Enfin, nous notons $Cut(C_k, A_i)$ le partitionnement en deux du jeu de données composé des documents inclus dans le cluster C_k et transformés avec l'ensemble des attributs A_i, en utilisant l'algorithme **SuSE** instancié pour rechercher deux clusters.

Les étapes principales de notre nouvelle méthode de clustering sont présentées par l'algorithme 12. Elle consiste à choisir à chaque étape la division d'intérêt maximum parmi l'ensemble des divisions possibles des clusters courants C_k pour $k \in [1, K]$ sur les ensembles d'attributs possibles $A_i \in A$, jusqu'à ce que le nombre attendu de clusters soit atteint. L'intérêt de partitionner les éléments d'un cluster C_k sur les attributs de A_i est évalué par le rapport entre la log-vraisemblance d'une partition avec deux clusters et la log-vraisemblance d'une partition avec un seul cluster, pondéré par le nombre de membres du cluster C_k pour préférer les divisions des clusters contenant davantage de documents. La sortie de la procédure est alors un arbre de décision dans lequel chaque nœud correspond à un test d'appartenance à une règle, créée comme présentée dans la section 5.7, et définie par un minimum d'attributs possible.

6.2.3 Classification supervisée de documents XML

Afin de bénéficier de la capacité de **SuSE** à présenter des résultats compréhensibles, nous proposons maintenant de l'adapter pour la classification supervisée de documents XML. La nouvelle méthode est alors composée de deux étapes principales : une première étape qui agit comme un clustering mais utilise les classes connues des objets, et une deuxième étape complètement guidée par les classes. La sortie de la méthode est alors un arbre de décision construit de manière à être aussi compréhensible que possible.

La première étape est décrite par l'algorithme 13 et consiste en une phase de clustering qui autorise la présence de plusieurs classes dans un même cluster mais ne permet pas qu'une classe soit séparée dans différents clusters. À cette étape, nous préférons qu'une règle de découpage soit définie sur des attributs de plus bas niveau qui sont plus *simples* que des attributs de plus haut niveau. Plus formellement, nous préférons utiliser A_i que A_j si $i < j$. C'est pourquoi nous effectuons une division dès que possible plutôt

Algorithme 12 SuSE pour XML.

Entrée : l'ensemble D de documents XML et le nombre attendu de clusters nb_clus.
  $K = 1$ {nombre courant de clusters de la partition}
  $C_1 = D$ {initialiser l'unique cluster C_1 avec tous les documents de D}
  $O = \emptyset$ {créer un nouvel arbre de décision vide O}
  while $K \neq nb\_clus$ **do**
    for all $k \in [1, K]$ **do**
      for all $i \in [1, SA]$ **do**
        calculer l'intérêt de $Cut(C_k, A_i)$
      end for
    end for
    sélectionner et exécuter la division d'intérêt maximum
    calculer la règle associée et l'ajouter à l'arbre de décision O
    $K = K + 1$
  end while
Sortie : l'arbre de décision O, avec $nb\_clus - 1$ tests.

que de comparer l'intérêt de plusieurs divisions possibles comme il est fait en clustering. Le même raisonnement sera utilisé à l'étape suivante.

La seconde étape de notre méthode est décrite par l'algorithme 14. Elle prend en entrée la sortie de l'étape précédente et consiste à séparer les exemples de classes différentes qui sont encore inclus dans les mêmes clusters. Elle est elle-même composée de deux étapes : la première essaye d'utiliser des règles pour distinguer les classes, dans le but d'être le plus compréhensible possible, tandis que la seconde étape utilise des modèles probabilistes, qui sont des modèles plus riches, capables de décrire des zones de séparation plus complexes.

1. Si une règle est trouvée qui est capable de distinguer une classe des autres, alors cette règle est utilisée comme test suivant dans l'arbre de décision. Comme nous l'avons expliqué précédemment, une division est exécutée dès que possible.

2. Puis si aucune règle capable de distinguer une classe des autres dans un cluster donné n'a été trouvée, alors nous testons l'erreur en validation croisée de modèles probabilistes générés sur chaque ensemble possible d'attributs de A, et celui qui a mené au taux d'erreur le plus faible est sélectionné comme test suivant dans l'arbre de décision.

Dans l'arbre de décision final, les tests sur chaque nœud peuvent donc être de deux natures différentes : ils peuvent correspondre à un test d'appartenance à une règle, ou bien à un test probabiliste sur des modèles probabilistes. Chacun de ces tests est exécuté sur seulement un ensemble d'attributs à la fois.

Algorithme 13 SuSE pour XML supervisé : étape 1.

Entrée : l'ensemble D de documents XML.

$K = 1$ {nombre courant de clusters de la partition}
$C_1 = D$ {initialiser l'unique cluster C_1 avec l'ensemble des documents de D}
$O = \emptyset$ {créer un nouvel arbre de décision vide O}
$CUT = 1$
while $CUT = 1$ **do**
  $CUT = 0$
  for all $k \in [1, K]$ **do**
    $CUT_k = 0$ et $i = 1$
    while $CUT_k = 0$ et $i <= SA$ **do**
      if (aucune classe n'a été séparée dans différents clusters par $Cut(C_k, A_i)$) **then**
        exécuter la division ; $K = K + 1$
        calculer la règle associée et l'ajouter à l'arbre de décision O
        $CUT_k = 1$ et $CUT = 1$
      else
        $i = i + 1$
      end if
    end while
  end for
end while

Sortie : l'arbre de décision O, et la partition courante $P = \{C_k | k \in [1, K]\}$.

Algorithme 14 SuSE pour XML supervisé : étape 2.

Entrée : l'arbre de décision O et la partition de l'étape précédente $P = \{C_k | k \in [1, K]\}$.

```
for all k ∈ [1, K] do
  while C_k contient plusieurs classes différentes do
    for all i ∈ [1, SA] do
      for all class ∈ C_k do
        if (une règle est capable de distinguer la classe des autres) then
          exécuter la division et mettre à jour l'arbre de décision O
        end if
      end for
    end for
    if (aucune division n'a été effectuée) then
      for all i ∈ [1, SA] do
        calculer l'erreur de classification en validation croisée du modèle probabiliste
        généré sur les attributs A_i
      end for
      choisir le modèle qui a mené au taux d'erreur le plus faible
      mettre à jour l'arbre de décision O
    end if
  end while
end for
```

Sortie : l'arbre de décision O.

6.3 Bilan

De manière générale, toute méthode peut être adaptée lorsque l'on possède certaines connaissances supplémentaires sur un domaine particulier. Dans le cas d'un apprentissage supervisé par exemple, si l'utilisateur sait que chaque classe peut être représentée par une unique distribution, alors parmi les modèles possibles que nous avons présentés dans la première section de ce chapitre, c'est le premier modèle qui sera utilisé. En pratique, puisqu'il est rare de posséder de telles informations, c'est souvent le modèle le plus simple qui est d'abord considéré, puis des modèles de plus en plus complexes tant que les résultats ne sont pas jugés satisfaisants. Une autre méthode étudiée dans [Biernacki and Govaert, 1998] consiste à tester un ensemble de modèles possibles et à utiliser un test statistique pour choisir celui qui est le plus approprié aux données.

Dans le cas de l'extension des méthodes aux données semi-structurées, nous verrons dans le chapitre suivant concernant nos expérimentations l'intérêt d'utiliser la connaissance que nous avons sur l'existence de plusieurs ensembles spécifiques d'attributs représentant différentes vues sur la structure des documents XML. Bien que la méthode de transformation des arbres XML en ensembles d'attributs-valeurs se traduise par une perte d'information, nous verrons qu'elle permet d'obtenir de très bons résultats en pratique. Nous pourrons également observer l'intérêt des adaptations de **SuSE** que nous avons proposées dans l'objectif de fournir un résultat compréhensible à l'utilisateur.

Chapitre 7

Expérimentations

Comme nous l'avons déjà évoqué dans la section 2.5, évaluer les résultats fournis par des algorithmes de clustering est un problème difficile car l'intérêt de ces résultats comporte une certaine part de subjectivité.

Nous avons présenté différentes méthodes utilisées classiquement pour effectuer cette évaluation, et nous avons mis en avant leurs limitations. Nous soutenons qu'évaluer les résultats d'algorithmes de clustering en choisissant l'une de ces méthodes n'est pas satisfaisant. Par contre, mener plusieurs de ces évaluations permet de mettre en avant plusieurs propriétés de l'algorithme que nous souhaitons évaluer.

En effet, démarrer par une analyse des résultats de l'algorithme face à des jeux de données générés artificiellement permet d'observer la robustesse de la méthode dans différentes conditions qui sont contrôlées. Confronter ensuite l'algorithme à des jeux de données réels pour lesquels un expert est capable de spécifier si les clusters ont du sens permet par ailleurs d'observer son intérêt dans le cas d'applications réelles.

Dans ce chapitre, nous confrontons donc les nouvelles méthodes que nous avons proposées à ces méthodes classiques d'évaluation. Puis dans la partie suivante, nous poursuivons leur évaluation dans le cadre d'une nouvelle méthodologie qui présente plusieurs avantages par rapport aux méthodes classiques.

7.1 Données artificielles

Le principe de l'évaluation d'algorithmes de clustering à partir de données artificielles est de se fixer soi-même des clusters dans l'espace de description des objets, d'appliquer les algorithmes sur les jeux de données produits automatiquement, puis de mesurer les correspondances entre les clusters initiaux et ceux qui ont été retrouvés par les algorithmes. Autrement dit, on se place dans un cadre où l'objectif à atteindre est connu, et où l'on souhaite évaluer si les algorithmes sont bien capables de retrouver les clusters présents dans les données. On parle dans ce cas d'*évaluation externe*.

Dans le cadre du subspace clustering que nous nous sommes fixé, nous souhaitons tout d'abord évaluer la capacité des méthodes à retrouver des clusters définis dans différents sous-espaces de l'espace original de description des objets. Les principales

questions abordées dans cette section sont ensuite les suivantes :

1. quelle est l'influence des paramètres de **Tuareg** et **SuSE** ?
2. comment se comportent ces méthodes selon le nombre d'objets, le nombre de dimensions, le nombre et les caractéristiques des clusters présents dans les jeux de données, comparées aux autres méthodes existantes de clustering ou de subspace clustering ?
3. et sont-elles robustes à l'ajout de dimensions non pertinentes, à la présence de bruit dans les données, ou de valeurs manquantes ?

Pour ces expériences, nous produisons donc automatiquement des problèmes, un problème étant caractérisé par les paramètres suivants :

- N le nombre d'objets du jeu de données,
- M_n et M_c le nombre de dimensions, respectivement numériques et catégorielles, décrivant les objets,
- L le nombre de clusters,
- C le nombre moyen de dimensions caractéristiques des clusters,
- e_m et e_M les écarts types minimum et maximum des coordonnées des objets appartenant à un même cluster par rapport à son centre de gravité et sur ses dimensions caractéristiques numériques,
- et p le pourcentage d'objets appartenant à un même cluster et partageant la même modalité sur ses dimensions caractéristiques catégorielles.

L points d'ancrage $(\overrightarrow{\mu_1}, ..., \overrightarrow{\mu_L})$ sont tirés aléatoirement dans l'espace de description à $M = M_n + M_c$ dimensions, et sont utilisés comme centroïdes initiaux des clusters $(C_1, ..., C_L)$ à générer. L'intervalle de définition des dimensions numériques est [0,100], et le nombre moyen de modalités sur les dimensions catégorielles est de 5. À chacun de ces clusters est associée une partie des N objets, et un sous-ensemble, de taille proche de C, des M dimensions constituant ses dimensions caractéristiques. Puis les coordonnées des objets appartenant à un cluster C_k sont générées selon une loi normale de centre μ_{kd} et d'écart type $e \in [e_m, e_M]$ sur toute dimension numérique d caractéristique de C_k, et avec une probabilité p d'être égales à la modalité μ_{kd} sur toute dimension catégorielle d caractéristique de C_k ; elles sont générées selon une loi uniforme dans l'espace de description des dimensions non caractéristiques.

Nous utilisons ensuite la mesure d'Entropie (définie dans la section 2.6) pour évaluer la correspondance entre les résultats fournis par les méthodes de clustering évaluées et le clustering résultat attendu. Cette mesure est minimale lorsque la correspondance est maximale.

7.1.1 Algorithme Tuareg

Nous utilisons dans un premier temps ces jeux de données artificiels afin d'évaluer l'algorithme **Tuareg** sur différents aspects :

1. quelle est l'influence des paramètres d'entrée de **Tuareg** ? et plus particulièrement, peut-on fixer des valeurs par défaut à ces paramètres ?

2. comment **Tuareg** se comporte-t-il face à différents types de jeux de données ? autrement dit, quelles sont les performances de **Tuareg** selon le nombre d'objets, le nombre de dimensions, le nombre et les caractéristiques des clusters présents dans les jeux de données, comparé à K-means ?
3. **Tuareg** résiste-t-il à l'ajout de dimensions non pertinentes, sur lesquelles les coordonnées des objets sont générées uniformément sur l'espace de description des objets ?

Nous utilisons donc tout d'abord des jeux de données artificiels pour tester l'influence des paramètres de **Tuareg** sur ses performances. Dans ces expériences, les données sont générées avec les paramètres suivants (**Tuareg** tel que nous l'avons décrit n'étant pas capable de prendre en compte les dimensions catégorielles, seules des dimensions numériques sont utilisées ici pour décrire les objets du jeu de données) :

- nombre d'objets $N = 500$,
- nombre de dimensions numériques $M_n = 20$,
- nombre de clusters $L = 5$,
- nombre moyen de dimensions caractéristiques des clusters $C = 2$,
- et écarts types minimum $e_m = 3$ et maximum $e_M = 4$.

Chacun des résultats que nous présentons correspond à une moyenne sur 100 générations de jeux artificiels et exécutions de la méthode.

Les expériences ont ainsi mis en avant le fait que les performances de **Tuareg** sont améliorées lorsque le paramètre *ALEA_DIV*, contrôlant la composante stochastique de **Tuareg**, passe de 1 à 2, ce qui montre l'intérêt de cette composante stochastique. Ensuite, les résultats restent stables même lorsque la valeur de ce paramètre est encore augmentée. Nous avons donc fixé *ALEA_DIV* à 2 par la suite. Nous avons de même fixé *NB_RUNS*, le paramètre contrôlant le nombre d'itérations de la méthode de divisions successives de base, à 7 expérimentalement.

Concernant Max_K, le paramètre utilisateur servant de borne supérieure sur le nombre de clusters générés lors d'une itération de la méthode, son influence dépend de la méthode choisie pour la fonction *AcceptDivision*, qui permet de décider si la division d'un cluster peut ou non être effectuée. Si celle-ci est basée sur la comparaison de l'inertie courante du cluster avec l'inertie maximale entre les deux nouveaux clusters générés, alors l'algorithme produit un nombre de clusters proche du nombre réel de clusters présents dans le jeu de données. Ce nombre plafonne à 7 dans nos expériences, pour 5 clusters présents dans les données, et ce même quand Max_K dépasse les 20. Si au contraire, la fonction *AcceptDivision* est basée sur l'inertie minimale, alors le nombre de clusters générés est proche de Max_K. Dans la suite, c'est donc l'inertie maximale qui sera utilisée dans la fonction *AcceptDivision* pour tester si une division doit ou non être acceptée. Max_K constituant une borne supérieure qui permet d'assurer de ne pas générer trop de clusters, mais qui n'est pas forcément atteinte, nous l'avons fixée à 10 pour la suite.

Dans un deuxième temps, nous utilisons ces jeux de données artificiels afin d'observer les performances de **Tuareg** en fonction des caractéristiques des jeux de données qui lui sont fournis en entrée.

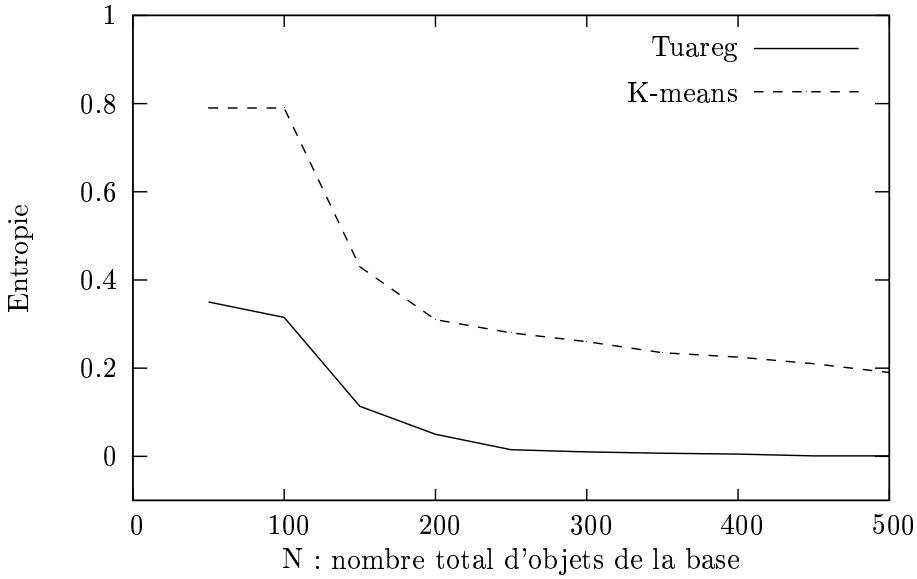
Afin de comparer les caractéristiques de **Tuareg** avec celles d'un algorithme de clustering très utilisé, et afin d'observer l'intérêt de la mise en œuvre de méthodes capables de prendre en compte le fait que certaines dimensions ne sont pas pertinentes pour la formation des clusters, nous avons choisi de comparer **Tuareg** à l'algorithme K-means : dans chaque cas, le nombre de clusters à trouver lui est fourni, les centroïdes sont initialisés de façon aléatoire, la méthode est exécutée dix fois, et le résultat minimisant l'inertie globale de la partition est conservé.

Les courbes de la figure 7.1 mettent alors en avant la robustesse de **Tuareg** face au nombre d'objets des jeux de données qui lui sont fournis en entrée. On observe sur la première courbe qu'à partir d'un certain nombre d'objets présents dans les données (200), les performances de **Tuareg** se stabilisent au dessus de celles de K-means, et proches de 0, ce qui dénote une correspondance maximale avec le clustering initial. La seconde courbe montre que le temps d'exécution de **Tuareg** reste raisonnable, même pour des jeux de données contenant beaucoup d'objets, puisque ses performances restent proches de celles de K-means, réputé pour son efficacité.

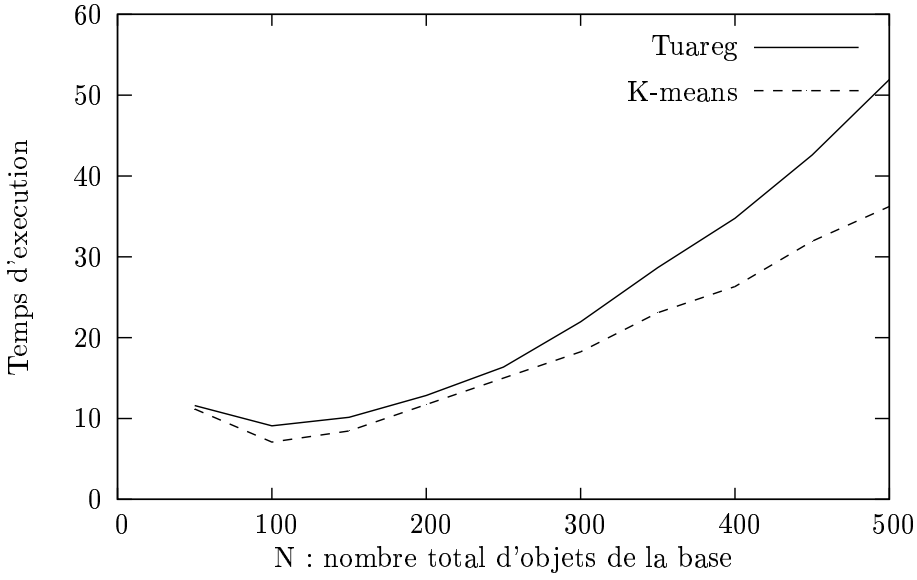
D'autres expériences ont été menées pour observer l'évolution des performances de **Tuareg** selon les caractéristiques des jeux de données qui lui sont fournis en entrée. Ainsi, il a été mis en avant que les performances de **Tuareg** restent stables face à M le nombre de dimensions du jeu de données, à L le nombre de clusters (tant qu'il n'est pas trop proche de Max_K), et à C le nombre moyen de dimensions caractéristiques des clusters.

Si le jeu de données est tel que pour tout couple de clusters distincts, il existe toujours au moins une dimension de projection sur laquelle ces deux clusters sont bien séparés (ce qui constitue l'hypothèse de base sur laquelle est basée **Tuareg**), alors les performances de **Tuareg** sont tout à fait satisfaisantes. Si cette hypothèse n'est pas vérifiée, les résultats restent cependant tout à fait raisonnables (avec une entropie autour de 0.2), parce qu'en produisant les objets d'un cluster suivant une loi normale centrée sur son centroïde, on rend plus probable la présence d'écarts importants aux marges de ce cluster plutôt qu'en son sein. De même, les résultats sont meilleurs lorsque l'écart type des coordonnées des membres de clusters par rapport à leur centroïde et sur leurs dimensions caractéristiques est faible (inférieur à 5), mais ils restent tout à fait raisonnables si cet écart type est plus important (avec une entropie à 0.2 pour un écart type à 10).

Par ailleurs, et contrairement à K-means, **Tuareg** reste stable face à la présence dans les jeux de données de dimensions non pertinentes sur lesquelles les valeurs des objets sont réparties selon une distribution uniforme. Ce résultat est illustré par les courbes de la figure 7.2, représentant l'Entropie et le temps d'exécution des méthodes en fonction du nombre M^+ de dimensions non pertinentes ajoutées dans les jeux de données. De plus, il apparaît que lorsque le nombre de dimensions non pertinentes devient important, alors **Tuareg** devient plus rapide que K-means parce qu'il nécessite alors de plus en plus d'itérations pour converger.

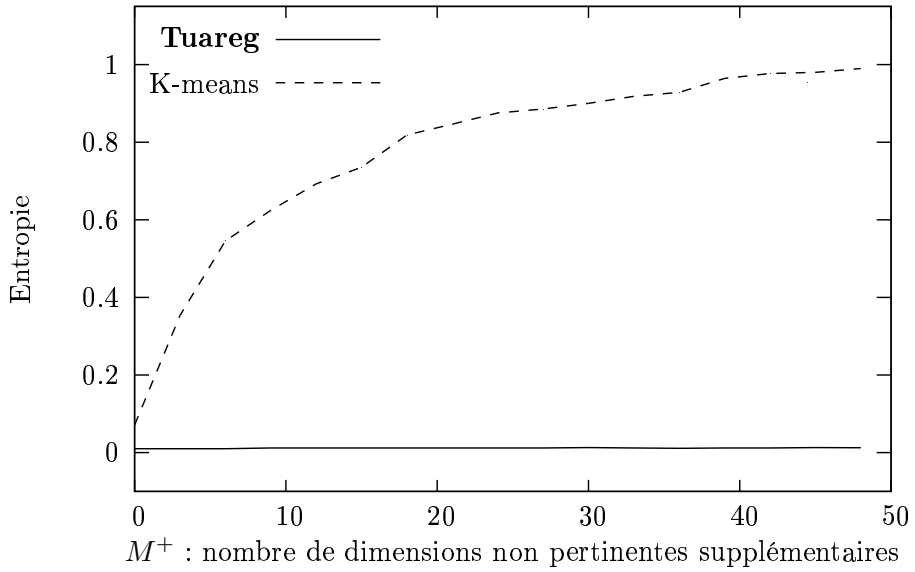


(a) Entropie : **Tuareg** surpasse K-means, et est proche de 0 quand $N > 200$.

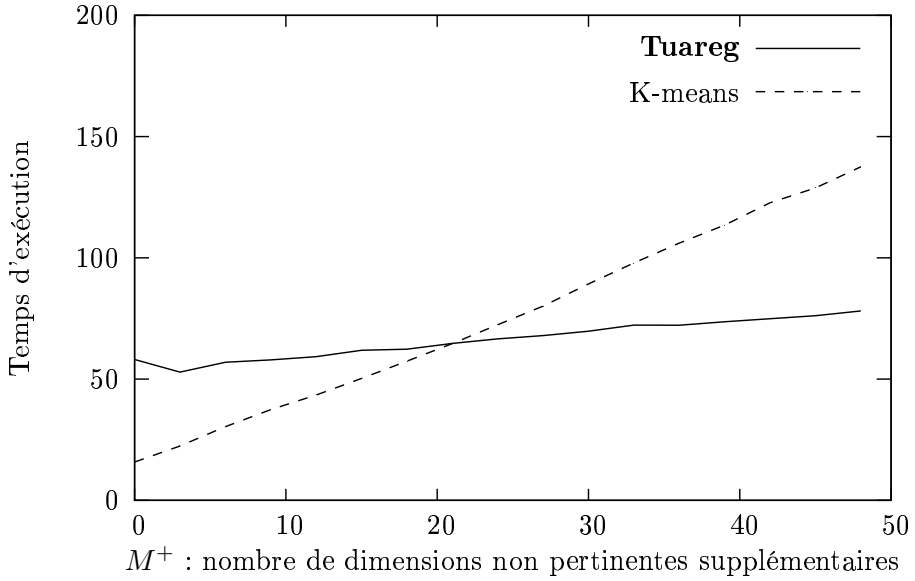


(b) Temps d'exécution : l'évolution pour **Tuareg** suit celle de K-means.

FIG. 7.1 – Performances de **Tuareg** en fonction du nombre d'objets présents dans les jeux de données.



(a) Entropie : **Tuareg** reste stable, contrairement à K-means.



(b) Temps d'exécution : **Tuareg** devient plus rapide que K-means quand $M^+ > 20$.

FIG. 7.2 – Performances de **Tuareg** face à la présence de dimensions non pertinentes dans les jeux de données.

7.1.2 Algorithme SuSE

De même que pour **Tuareg**, nous utilisons des jeux de données artificiels afin d'évaluer l'algorithme **SuSE** sur différents aspects :

1. quelle est l'influence des paramètres d'entrée de **SuSE** ?
2. comment **SuSE** se comporte-t-il selon le nombre d'objets, le nombre de dimensions, le nombre et les caractéristiques des clusters présents dans les jeux de données, comparé à K-means et LAC, une adaptation de K-means au subspace clustering ?
3. **SuSE** résiste-t-il à l'ajout de dimensions non pertinentes, et à la présence de bruit dans les données ?

Nos premières expérimentations concernent l'influence des paramètres de **SuSE** sur ses performances. Rappelons que **SuSE** est basé sur l'utilisation de deux paramètres : K le nombre de clusters présents dans les données, et R le nombre de dimensions pertinentes des clusters.

La figure 7.3 montre tout d'abord, sur un exemple simplifié, l'influence du paramètre K sur les résultats de **SuSE**. On peut ainsi observer que lorsque le nombre de clusters recherchés par **SuSE** est inférieur au nombre réel, alors quelques concepts sont fusionnés mais le résultat ne s'éloigne pas complètement de la solution réelle. À l'inverse, lorsque le nombre de clusters recherchés est supérieur au nombre réel, alors plusieurs concepts se recouvrent.

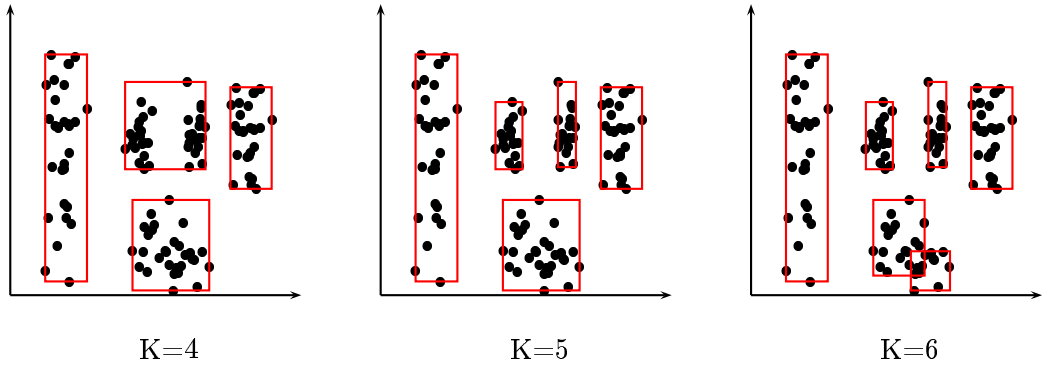


FIG. 7.3 – Exemple d'influence du paramètre K de **SuSE**.

La suite de ces expérimentations sur **SuSE** consiste à valider l'intérêt de l'utilisation du critère BIC pour déterminer automatiquement les valeurs les plus appropriées pour ces deux paramètres K et R. Les paramètres des jeux de données utilisés pour ces expérimentations sont les suivants :

– nombre d'objets $N \in [50, 1000]$,

- nombre de dimensions $M \in [10, 200]$,
- nombre de clusters $L \in [2, 6]$,
- nombre moyen de dimensions caractéristiques des clusters $C \in [2, 10]$,
- écarts types minimum $e_m = 3$ et maximum $e_M = 9$,
- et pourcentage d'objets partageant la même modalité $p = 0.8$.

Chacun des résultats que nous présentons correspond à une moyenne sur 100 générations de jeux artificiels et exécutions de la méthode.

La figure 7.4 montre ainsi l'évolution de la valeur du critère BIC en fonction du nombre R de dimensions sélectionnées pour chaque cluster lorsque le nombre K de clusters présents dans les données est fourni à l'algorithme. Cette courbe met alors en avant le fait que la valeur du critère BIC décroit jusqu'à ce que R atteigne C, le nombre réel de dimensions pertinentes associées aux clusters, puis augmente quand R devient plus important que C. Cela montre donc expérimentalement que le critère BIC peut être utilisé pour déterminer automatiquement le nombre approprié de dimensions à associer à chaque cluster.

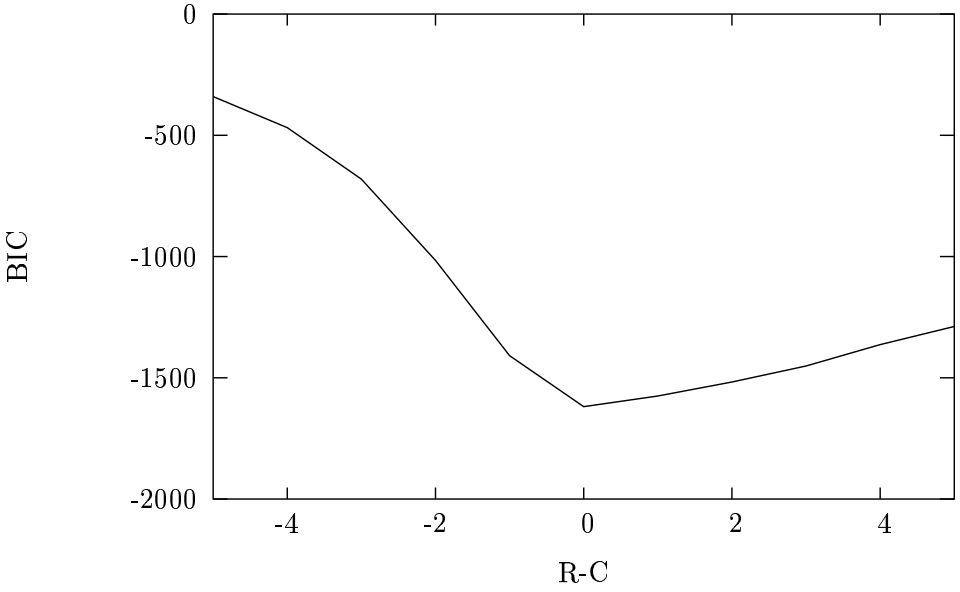


FIG. 7.4 – Évolution de la valeur du critère BIC en fonction du nombre R de dimensions sélectionnées pour chaque cluster.

De la même façon, la figure 7.5 montre l'évolution de la valeur du critère BIC en fonction du nombre K de clusters recherchés et du nombre R de dimensions pertinentes associées aux clusters. Pour une meilleure visualisation, -BIC est reporté au lieu de BIC. Cela montre donc expérimentalement que la valeur optimale du critère BIC est atteinte lorsque K atteint L et que R atteint C, ce qui montre bien l'intérêt d'utiliser un tel critère statistique afin de déterminer automatiquement les paramètres (inhérents à toute méthode de subspace clustering) les plus appropriés aux données. Par ailleurs, ces

résultats suggèrent l'utilisation d'une méthode de gradient pour atteindre cette valeur optimale du critère BIC.

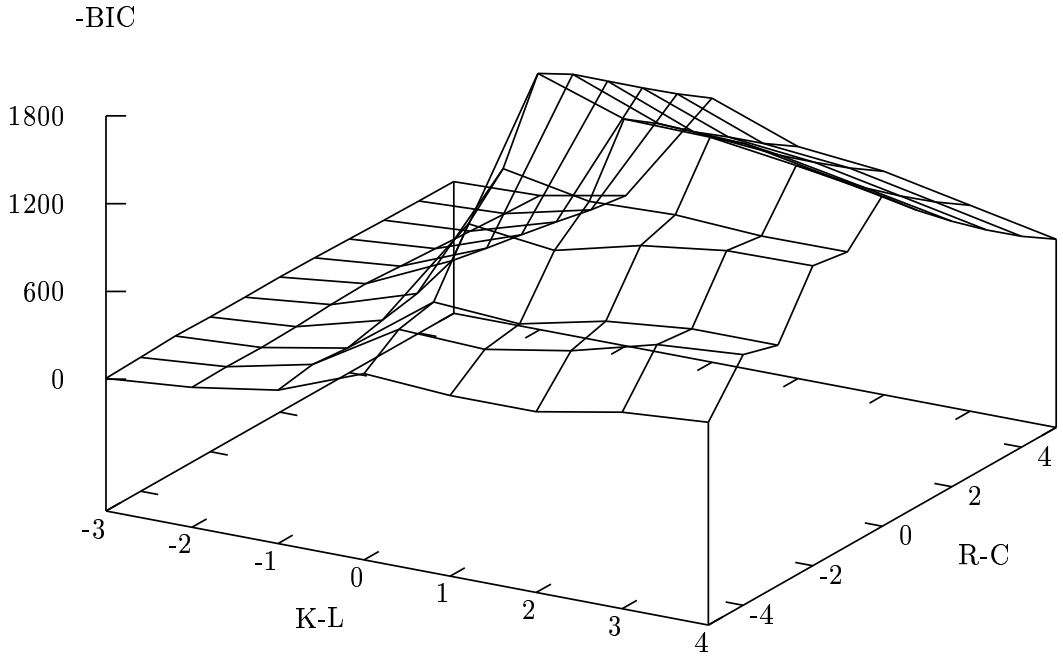


FIG. 7.5 – Évolution de la valeur du critère BIC en fonction du nombre K de clusters recherchés et du nombre R de dimensions sélectionnées pour chaque cluster.

Dans la suite de ces expérimentations visant à évaluer la robustesse de **SuSE** face à différents types de jeux de données, nous proposons de considérer que les données sont décrites par des attributs numériques uniquement, car cela nous permet de comparer **SuSE** aux méthodes existantes de subspace clustering.

Notre méthode s'apparente à la famille des méthodes de subspace clustering descendantes sur les dimensions. Parmi les plus récentes, LAC [Domeniconi et al., 2004], détaillé en annexes B.3.5, est une méthode efficace qui nécessite un seul paramètre de la part de l'utilisateur : le nombre de clusters recherchés. Nous proposons donc de nous comparer à cet algorithme.

LAC est basé sur l'algorithme K-means et associe à chaque centroïde de cluster un vecteur de poids sur chaque dimension de l'espace. À chaque étape, ces vecteurs sont mis à jour en fonction de la dispersion des objets du cluster sur les dimensions : plus la dispersion est élevée, plus le poids associé à la dimension est faible.

La figure 7.6 illustre les résultats de notre méthode est ceux de LAC face à un jeu de données généré artificiellement. Sur cet exemple, nous pouvons observer une limitation classique des méthodes de type K-means face aux méthodes probabilistes : les premières recherchent des clusters de forme hypersphérique alors que les secondes sont capables d'identifier des clusters de forme allongée et permettent naturellement le chevauchement des clusters.

(a) LAC (b) **SuSE**

FIG. 7.6 – LAC versus **SuSE** sur l'exemple.

Nous comparons également **SuSE** à **SSC**, la méthode de clustering statistique également basée sur l'hypothèse d'indépendance des dimensions, mais qui ne met pas en œuvre la sélection des dimensions les plus pertinentes associées aux clusters. Le nombre de clusters recherchés pour chacun des problèmes est fourni à l'ensemble des méthodes évaluées. Et le paramètre Max_R de **SuSE** dénotant le nombre maximum de dimensions pertinentes associées aux clusters est fixé à 10.

La figure 7.7 montre tout d'abord le temps d'exécution de différents algorithmes en fonction de l'évolution du nombre d'objets présents dans les jeux de données. Dans ce cas, SSC1 se réferre à la méthode de clustering statistique n'utilisant pas la méthode d'arrêt des itérations de type K-means, alors que SSC2 se réferre à la méthode qui utilise ce critère d'arrêt. On peut alors observer l'intérêt d'utiliser ce critère d'arrêt, puisque, pour des résultats de qualité similaire, le temps d'exécution de la méthode est nettement amélioré, plus proche de celui de LAC que de celui de la méthode statistique classique.

Le tableau 7.1 présente ensuite les moyennes, sur 500 générations aléatoires de jeux de données, des entropies et temps d'exécutions de K-means, LAC, **SSC** et **SuSE**. Il nous permet ainsi d'observer que **SSC** et **SuSE** ont des résultats similaires sur jeux de données artificiels, supérieurs à ceux de LAC et bien supérieurs à ceux de K-means. Au niveau du temps d'exécution, K-means n'est pas le plus rapide car, comme le modèle qu'il utilise n'est pas approprié aux données, il nécessite de nombreuses itérations avant de converger. LAC est plus rapide que **SSC**, mais c'est **SuSE** qui est le plus rapide, car contrairement aux autres méthodes, il ne considère qu'un sous-ensemble des dimensions de description pendant l'apprentissage. Ceci constitue l'un de ses avantages.

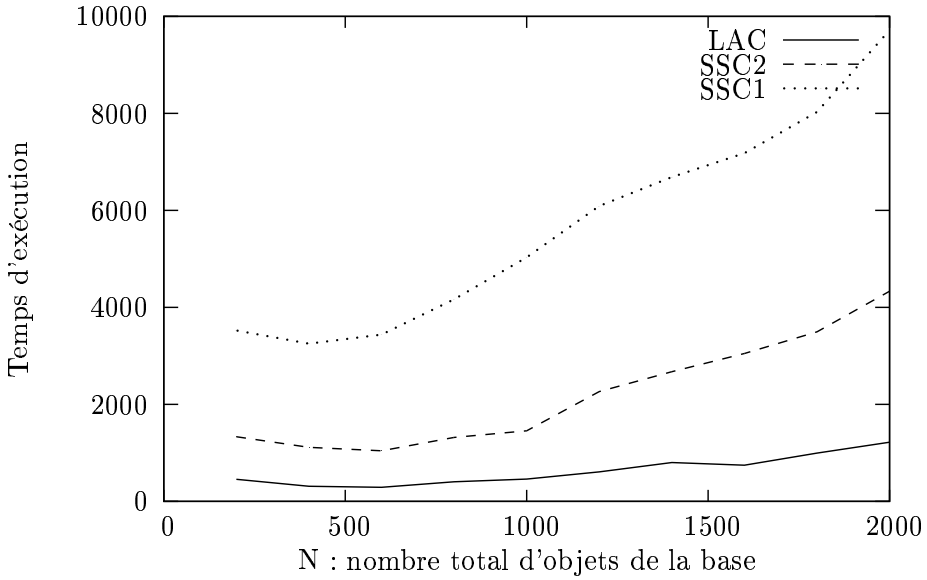


FIG. 7.7 – Intérêt du critère d'arrêt de type K-means dans **SSC**.

	K-means	LAC	**SSC**	**SuSE**
Entropie	0.974	0.434	0.263	0.263
Temps d'exécution	236	197	239	156

TAB. 7.1 – Entropie et temps d'exécution des méthodes sur jeux de données artificiels.

Les figures suivantes montrent enfin les évolutions des entropies et temps d'exécution de K-means, LAC, **SSC** et **SuSE** en fonction du nombre d'objets N, du nombre de dimensions M, et du pourcentage de bruit présents dans les jeux de données. Chaque mesure correspond à une moyenne sur 100 générations de jeux de données artificiels.

Sur la figure 7.8, on peut ainsi observer que les résultats des méthodes de subspace clustering se stabilisent à partir d'un certain nombre d'objets présents dans les jeux de données (300 pour **SSC** et **SuSE**, 200 pour LAC), alors que K-means continue à améliorer ses résultats lorsque davantage d'objets sont fournis en entrée de l'algorithme. Lorsque peu d'objets sont présents dans les jeux de données, alors **SuSE** surpasse légèrement **SSC**. Enfin, le temps d'exécution des méthodes augmente linéairement en fonction du nombre d'objets présents dans les données, mais celui de K-means augmente plus rapidement que les autres. **SuSE** possède le temps d'exécution le plus faible.

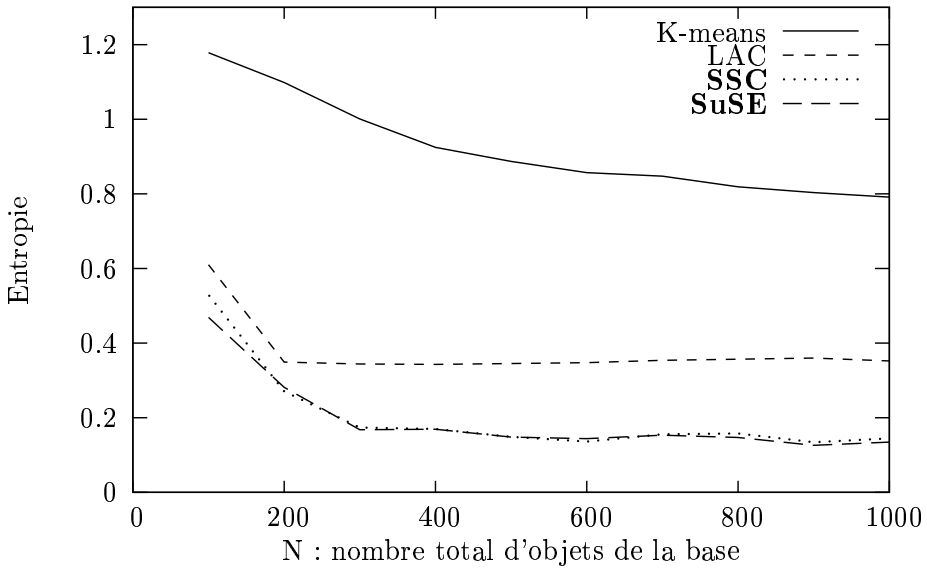
Sur la figure 7.9, on peut observer que les résultats des méthodes se dégradent lorsque le nombre de dimensions présentes dans les jeux de données augmente. Rappelons que le nombre moyen de dimensions pertinentes des clusters, lui, n'augmente pas. Davantage de dimensions de description induit donc également davantage de dimensions non pertinentes. Cependant, **SSC** et **SuSE**, qui partagent la même évolution, restent supérieurs à LAC et K-means. Par ailleurs, les temps d'exécution de **SSC** et **SuSE** augmentent moins rapidement que ceux de K-means et LAC lorsque le nombre de dimensions considérées augmente. **SuSE** possède le temps d'exécution le plus faible lorsque plus de 30 dimensions sont utilisées pour décrire les objets, ce qui met clairement en avant que son atout se situe au niveau du nombre de dimensions considérées pendant l'apprentissage.

La figure 7.10 reporte finalement l'évolution de l'entropie et du temps d'exécution des méthodes en fonction du pourcentage de bruit présent dans les jeux de données. On observe dans ce cas que **SSC** et **SuSE** sont beaucoup plus robustes que LAC et K-means à la présence de bruit dans les données, puisque l'écart en entropie entre ces deux ensembles de méthodes devient de plus en plus important lorsque le pourcentage de bruit dans les données augmente. Le temps d'exécution des méthodes augmente linéairement en fonction du bruit, plus rapidement lorsque les premiers objets bruités apparaissent dans les jeux de données.

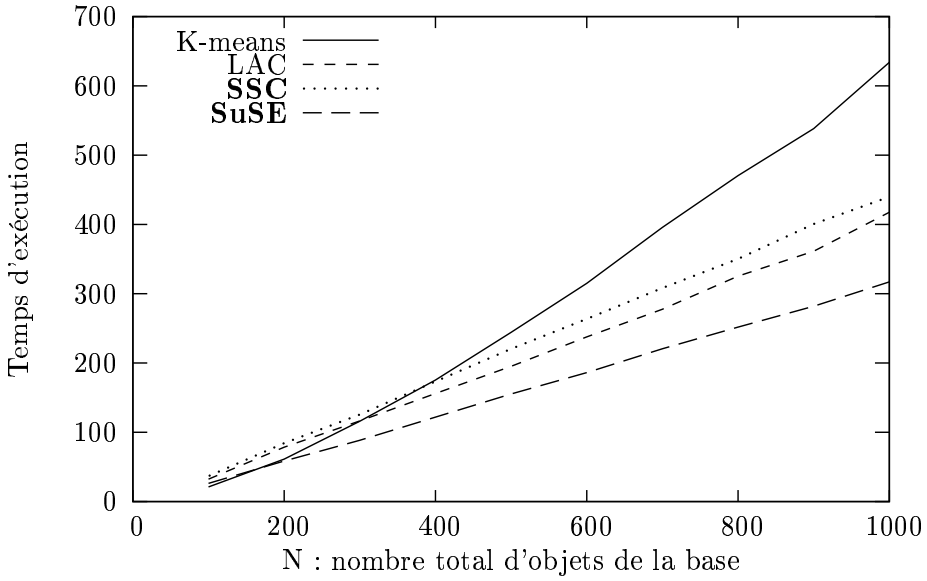
Enfin, notons que les résultats de notre algorithme sont tout aussi robustes si les données artificielles sont générées selon des lois uniformes dans des intervalles de définition spécifiques aux clusters sur leurs dimensions caractéristiques numériques, au lieu de lois gaussiennes. Autrement dit, un modèle gaussien est également approprié pour approximer des clusters distribués de façon uniforme dans certaines zones spécifiques de l'espace de description des objets. Les résultats sont également robustes à la présence de certaines valeurs manquantes dans les jeux de données.

7.1.3 Présentation des résultats

Nous verrons dans la section suivante portant sur la confrontation de nos algorithmes à des données réelles que **SuSE** obtient de meilleurs résultats que **Tuareg**, précisément parce que dans de nombreux cas réels, bien que les clusters présents dans les données soient séparés dans certains sous-espaces de l'espace original de description des objets,

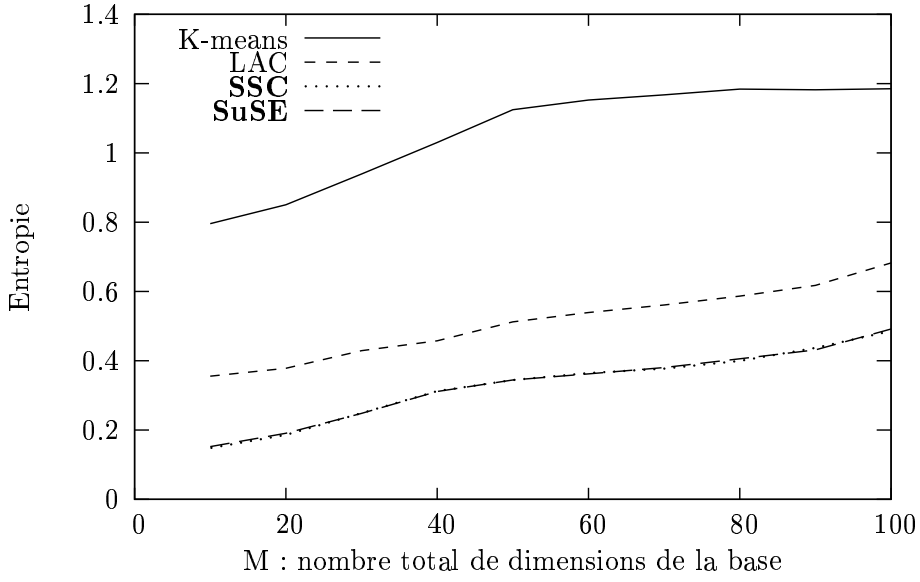


(a) Entropie : **SSC** et **SuSE** surpassent K-means et LAC et se stabilisent pour $N > 300$.

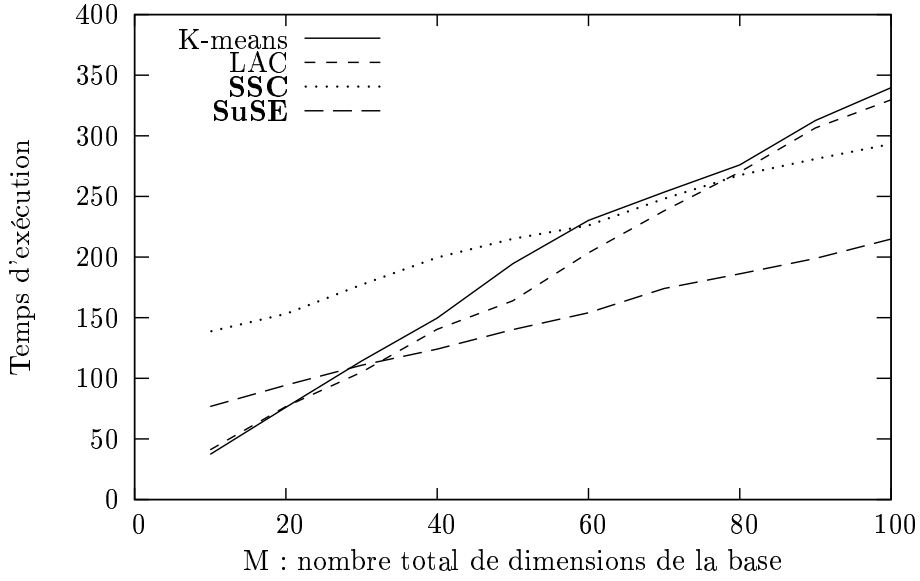


(b) Le temps d'exécution des méthodes augmente linéairement en fonction de N, plus rapidement pour K-means.

FIG. 7.8 – Performances des méthodes en fonction du nombre d'objets présents dans les jeux de données.

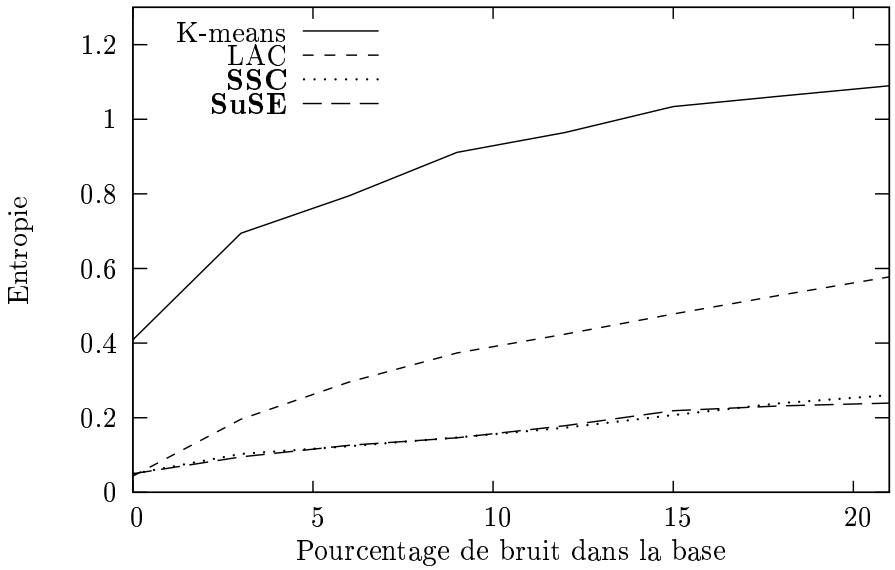


(a) Entropie : **SSC** et **SuSE** surpassent K-means et LAC.

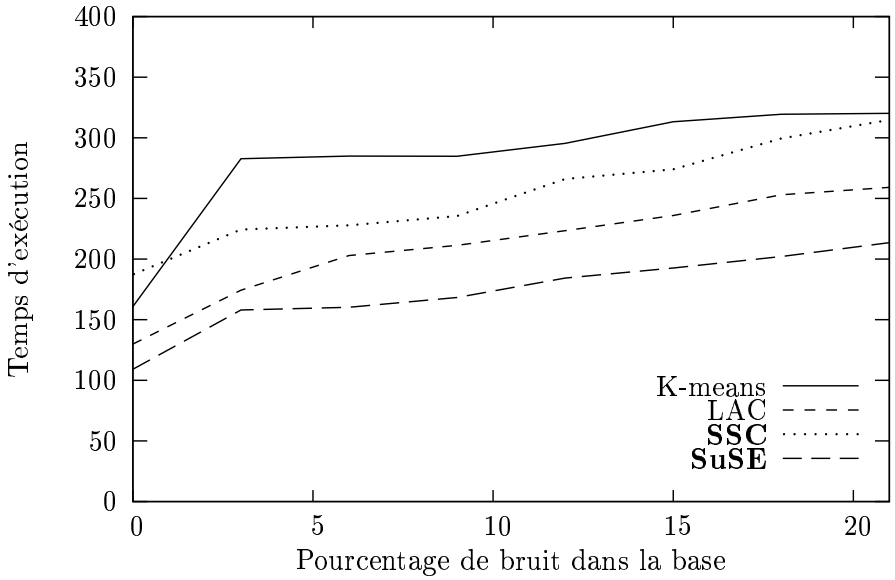


(b) Le temps d'exécution des méthodes augmente linéairement en fonction de M, plus rapidement pour K-means et LAC. **SuSE** est le plus rapide lorsque $M > 30$.

FIG. 7.9 – Performances des méthodes en fonction du nombre de dimensions présentes dans les jeux de données.



(a) Entropie : **SSC** et **SuSE** sont plus robustes au bruit que K-means et LAC.



(b) Le temps d'exécution des méthodes augmente linéairement en fonction du bruit, plus rapidement à l'apparition des premiers objets bruités.

FIG. 7.10 – Performances des méthodes en fonction du pourcentage de bruit présent dans les jeux de données.

il n'existe aucune dimension unique de projection sur laquelle ces clusters sont bien séparés. Or c'est justement sur l'hypothèse de l'existence de telles séparations unidimensionnelles qu'est basée **Tuareg**.

Par contre, **Tuareg** et **SuSE** sont tous deux basés sur l'utilisation de règles pour décrire les clusters identifiés. Avant d'observer l'intérêt de telles représentations sur données réelles, nous présentons ici sur un exemple artificiel l'efficacité de la méthode de description minimale des clusters que nous avons proposée, et de la méthode de visualisation graphique des résultats. Pour cela, nous générons un exemple de jeu de données artificiel ayant les caractéristiques suivantes :

- nombre d'objets $N = 300$,
- nombre de dimensions numériques $M_n = 10$,
- nombre de dimensions catégorielles $M_c = 5$,
- nombre de clusters $L = 3$,
- nombre moyen de dimensions caractéristiques des clusters $C = 4$,
- écarts types minimum $e_m = 2$ et maximum $e_M = 5$,
- et pourcentage d'objets partageant la même modalité $p = 0.8$.

Un jeu de données obtenu ainsi est décrit sous forme de règles comme suit :

- $C_0(N_0 = 104) \Rightarrow d_0 \in [64, 87], d_2 \in [84, 104], d_8 \in [27, 47], d_{13} = D$ et $d_{14} = D$
- $C_1(N_1 = 66) \Rightarrow d_5 \in [25, 43]$ et $d_8 \in [51, 68]$
- $C_2(N_2 = 130) \Rightarrow d_2 \in [55, 79], d_4 \in [75, 92], d_9 \in [69, 92]$ et $d_{14} = E$

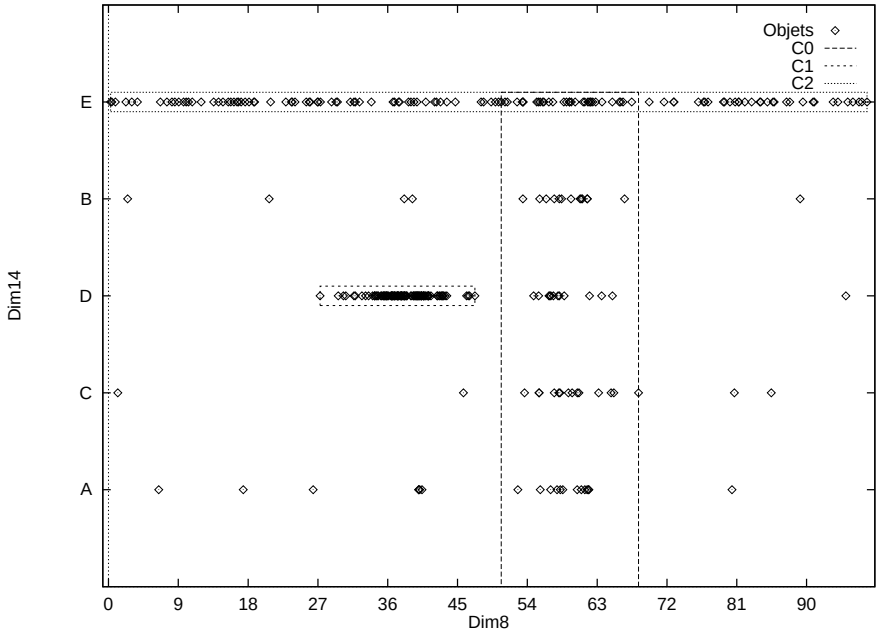
Sur ce jeu de données, la représentation sous forme de règles des clusters ciblés est la suivante :

- $C_0 \Rightarrow d_2 \in [84, 104], d_{10} = A$ et $d_{14} = D$
- $C_1 \Rightarrow d_5 \in [25, 43]$ et $d_8 \in [51, 68]$
- $C_2 \Rightarrow d_2 \in [55, 79], d_4 \in [75, 92]$ et $d_{14} = E$

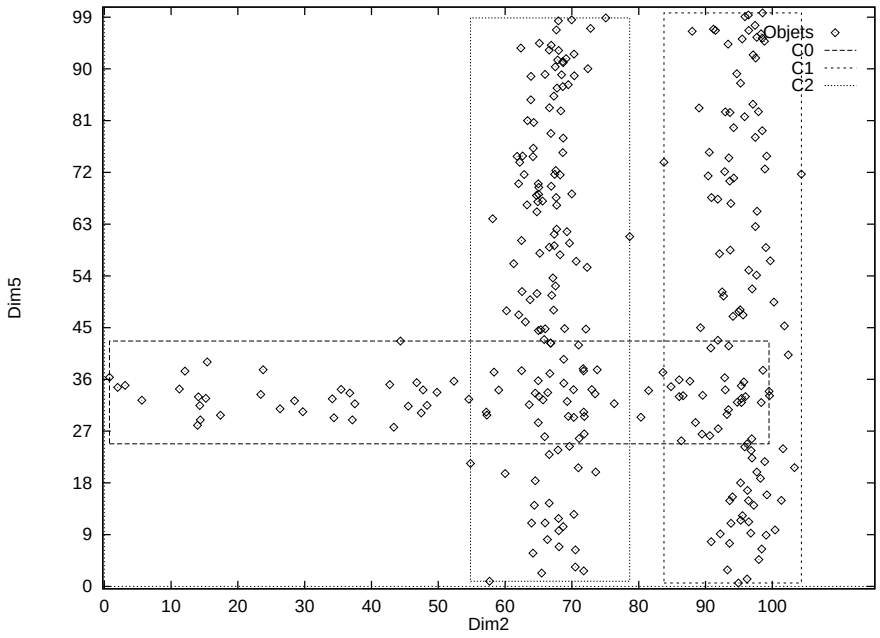
Nous observons alors que notre méthode de description minimale des clusters nous permet de réduire efficacement le nombre de dimensions décrivant les clusters, puisque certaines dimensions utilisées initialement dans la description des clusters ont pu être ignorées dans leur description finale.

Les visualisations graphiques correspondant aux deux couples de dimensions dont le poids est maximum sont fournies par la figure 7.11, mettant ainsi en avant l'intérêt de telles visualisations. En ce qui concerne la première projection, les objets qui ne semblent affectés à aucun cluster correspondent en fait aux objets qui ne partagent pas la catégorie la plus probable des clusters sur la dimension catégorielle d_{14}.

Par ailleurs, la figure 7.12 montre l'intérêt de prendre en compte le bruit qui peut exister dans les données afin de fournir un résultat aussi compréhensible que possible. Les points isolés sont alors ceux qui ont une probabilité plus forte d'appartenir à un cluster réparti uniformément dans l'espace de description des objets que d'appartenir aux deux clusters identifiés (cf. section 5.7). On peut ainsi observer sur cet exemple que le cluster C_1 est beaucoup mieux identifié après que les objets bruités aient été supprimés.

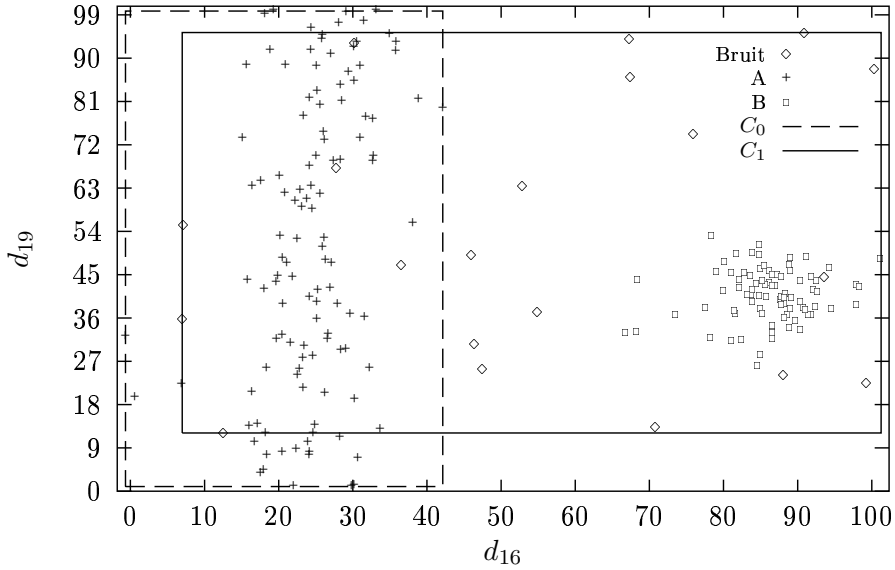


(a) Projections sur les dimensions d_8 et d_{14}.

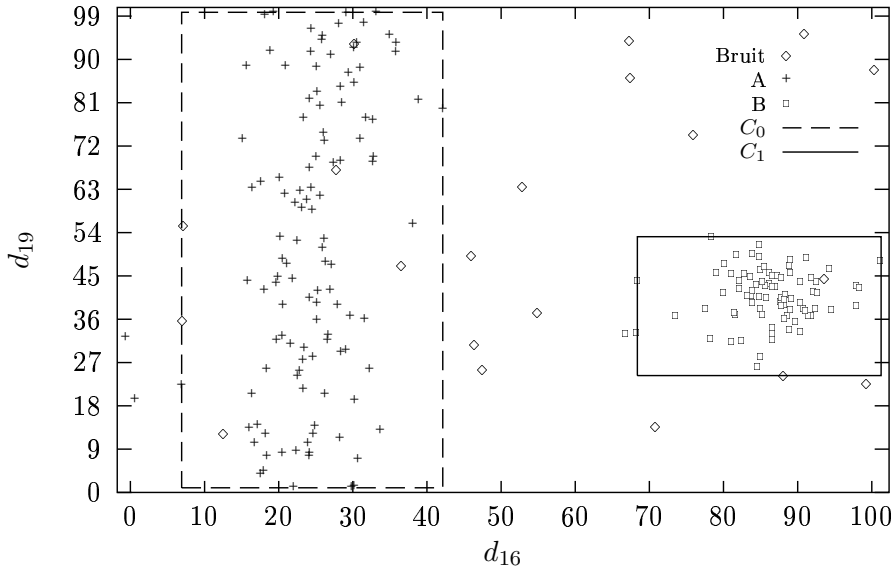


(b) Projections sur les dimensions d_2 et d_5.

FIG. 7.11 – Visualisation graphique des clusters obtenus par **SuSE** sur l'exemple.



(a) **SuSE** sans détection du bruit.



(b) **SuSE** avec détection du bruit.

FIG. 7.12 – Intérêt de la détection du bruit existant dans les données.

7.2 Données réelles

La deuxième partie de ces expérimentations porte sur la confrontation des méthodes de subspace clustering **Tuareg** et **SuSE** que nous avons proposées à des données réelles. Les premiers jeux de données considérés sont ceux issus de l'UCI Machine Learning Repository [Blake and Merz, 1998], qui sont souvent utilisés comme références. La plupart de ces jeux de données sont étiquetés, et dans de tels cas l'attribut de classe sera ignoré. Certains ne contiennent au contraire aucune information de classe et sont donc tout à fait appropriés à l'apprentissage non supervisé. Nous présentons ensuite quelques résultats obtenus face à des données réelles fournies par la société Pertinence. Là encore, il s'agit de jeux de données étiquetés pour lesquels l'attribut de classe est ignoré lors de l'apprentissage non supervisé. Dans certains cas, la méthode exhibe alors certaines informations sur les classes, mais il peut également arriver qu'elle mette en avant certains sous-concepts présents dans les données mais non liés aux classes du problème. Enfin, nous reportons les résultats que nous avons obtenus lors du challenge INEX portant sur la classification de documents semi-structurés, issus de bases de données réelles.

7.2.1 Données UCI

Le tout premier jeu de données que nous avons considéré pour tester nos méthodes est celui des *quadrupèdes*, disponible sur le site de l'UCI. Il s'agit d'un jeu de données dans lequel sont mélangées les descriptions de chevaux, de girafes, de chiens et de chats, tous caractérisés par les mesures de leurs membres (tête, cou, torse, queue, et quatre pattes), chaque membre étant vu comme un cylindre caractérisé par neuf attributs. Au total, un exemple est donc décrit par 72 attributs.

Lorsqu'un minimum de 70 animaux sont considérés, **Tuareg** obtient les résultats suivants : en plus de retrouver les quatre groupes pertinents, et ce sans connaître le nombre de groupes recherchés, **Tuareg** caractérise chaque groupe par des dimensions qui lui sont propres : les girafes sont caractérisées par leurs très longues jambes, les chevaux par leurs jambes de taille moyenne, les chiens par la petite taille de leur queue, et les chats par la petite taille de leur torse.

Nous obtenons donc bien un ensemble de groupes aux dimensions spécifiques, et ce résultat reste stable face à l'ajout de dimensions sur lesquelles les coordonnées de tous les objets sont tirées aléatoirement dans [0,100]. En effet, alors que les résultats de K-means se dégradent dès l'ajout d'une dimension aléatoire, **Tuareg** continue à retrouver les quatre groupes pertinents même après l'ajout de 1000 dimensions aléatoires.

SuSE a lui aussi de très bonnes performances sur ce jeu de données des quadrupèdes. Par contre, sur un autre jeu de données bien connu de l'UCI, les *Iris*, **Tuareg** obtient de moins bons résultats que **SuSE**. Dans ce jeu de données, 150 iris (des fleurs) sont décrits par quatre attributs et séparés en trois classes. L'une des classes (setosa) est bien séparée des deux autres (versicolor et virginica), mais ces dernières ne sont pas linéairement séparables, et **Tuareg** ne retourne donc que deux clusters. Il atteint là l'une de ses limites que nous avons présentées dans la figure 4.7 de la section 4.6. Par contre, **SuSE** identifie bien trois clusters présents dans les données. La meilleure projection en

deux dimensions des règles fournies en sortie est présentée dans la figure 7.13.

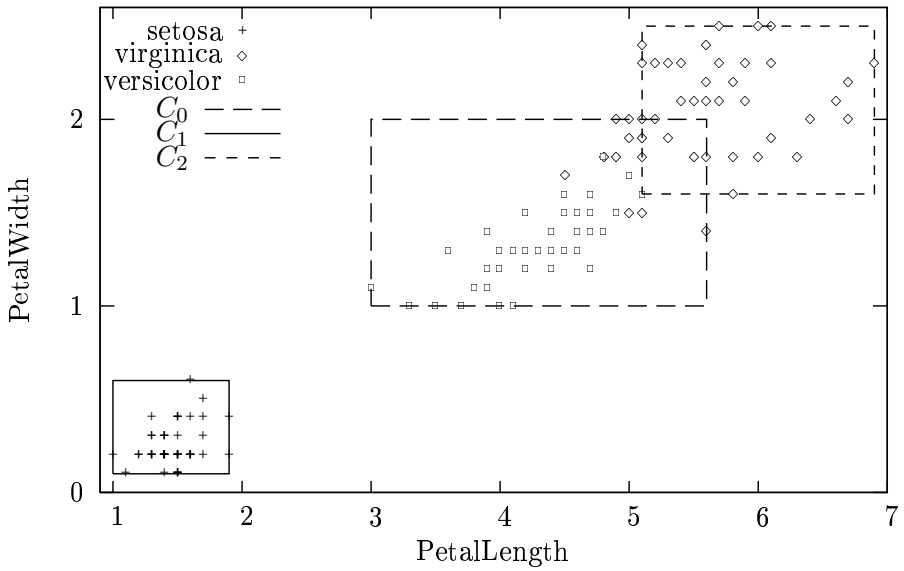


FIG. 7.13 – Visualisation graphique des résultats de **SuSE** sur le jeu de données Iris.

SuSE fournit des meilleurs résultats que **Tuareg** sur données réelles, non seulement parce que, contrairement à **Tuareg**, il est capable d'identifier des clusters séparés dans certains sous-espaces de l'espace original de description des objets mais dont la séparation n'est visible sur aucune projection à une seule dimension de ces objets, mais aussi parce qu'il est capable d'identifier naturellement des clusters qui se chevauchent. Par ailleurs, il est également capable de s'attaquer à des problèmes contenant des attributs catégoriels, alors que **Tuareg** ne l'est pas. C'est donc uniquement **SuSE** qui sera utilisé dans la suite de ces expérimentations.

La figure 7.14 montre une projection en deux dimensions obtenue en exécutant **SuSE** sur le jeu de données *wdbc* (*Wisconsin Breast Cancer Databases*), contenant 699 descriptions de tumeurs caractérisées par 30 attributs numériques, et séparées en 2 classes : M (malin) et B (bénin). L'un des deux clusters ciblés contient 393 objets de la classe B, définis par des valeurs faibles sur la majorité des dimensions de description, et l'autre cluster contient alors 80% de membres de la classe M. Nous sommes donc là en présence d'un cas où les concepts identifiés par **SuSE** sont très proches des classes du problème considéré (rappelons que les attributs de classe n'ont pas été utilisés lors de l'apprentissage).

Les résultats suivants que nous présentons concernent le jeu de données *Auto-Mpg* de l'UCI, qui contient la description d'un ensemble de 398 moteurs de voitures décrits par 8 attributs numériques (nombre de cylindres, de chevaux, accélération, etc.), mais aucun attribut de classe. Le tableau 7.2 présente les règles obtenues par **SuSE** avant leur simplification.

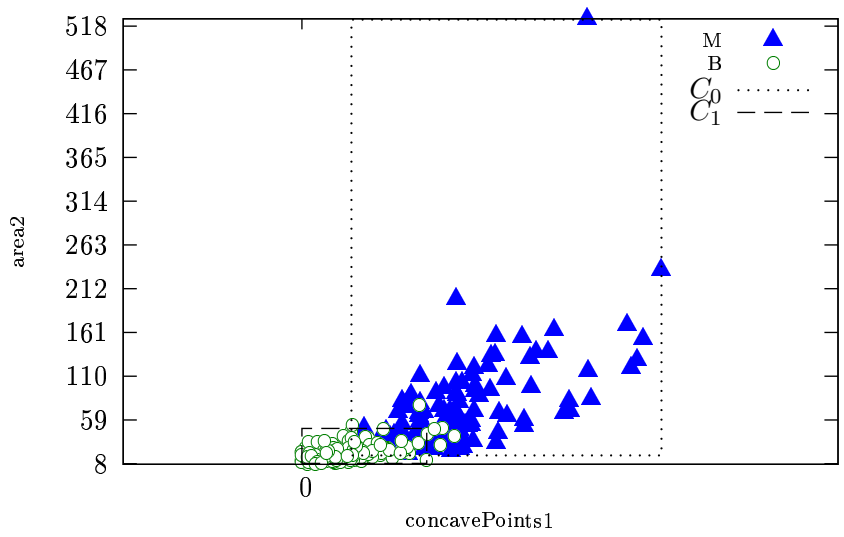


FIG. 7.14 – Visualisation graphique des résultats de **SuSE** sur le jeu de données wdbc.

mpg	cylinders	displacement	horsepower	weight	acceleration	modelYear	origin
[18,47]	[3,4]	[68,156]	[46,115]	[1613,3270]	[12,25]	[70,82]	[1,3]
[15,38]	[5,6]	[121,262]	[67,165]	[2472,3907]	[11,21]	[70,82]	[1,3]
[9,27]	[8,8]	[260,455]	[90,230]	[3086,5140]	[8,22]	[70,81]	[1,1]

TAB. 7.2 – Règles obtenues par **SuSE** sur le jeu de données Auto-Mpg.

Après la procédure de description minimale des règles trouvées, seul l'attribut *cylinders* est conservé pour caractériser l'appartenance des objets aux clusters. La figure 7.15 présente les visualisations graphiques associées aux clusters identifiés, qui mettent en avant les différences entre ces clusters, et la pertinence du résultat produit.

Enfin, des expériences ont également été menées sur des jeux de données contenant davantage d'attributs catégoriels, parmi lesquels le jeu de données *Automobile*, qui contient la description de 160 voitures caractérisées par 15 attributs numériques et 11 attributs catégoriels, et qui ne contient pas non plus d'attribut de classe. La figure 7.16 présente les visualisations graphiques associées aux résultats obtenus par **SuSE** sur ce jeu de données.

L'algorithme met ainsi en avant que le prix des voitures augmente fortement lorsque leur longueur dépasse les 170 (figure 7.16(a)), que les voitures ayant une traction arrière (*rwd*) ont un poids à vide supérieur aux tractions avant et quatre roues motrices (figure 7.16(b)), et que la majorité des voitures les plus chères sont à traction arrière (correspondance entre les deux figures concernant le cluster C_2). Là encore, les objets qui ne semblent affectés à aucun cluster dans la figure 7.16(b) correspondent en fait aux objets qui ne partagent pas la catégorie la plus probable des clusters sur la dimension catégorielle *drive-wheels* (*traction*).

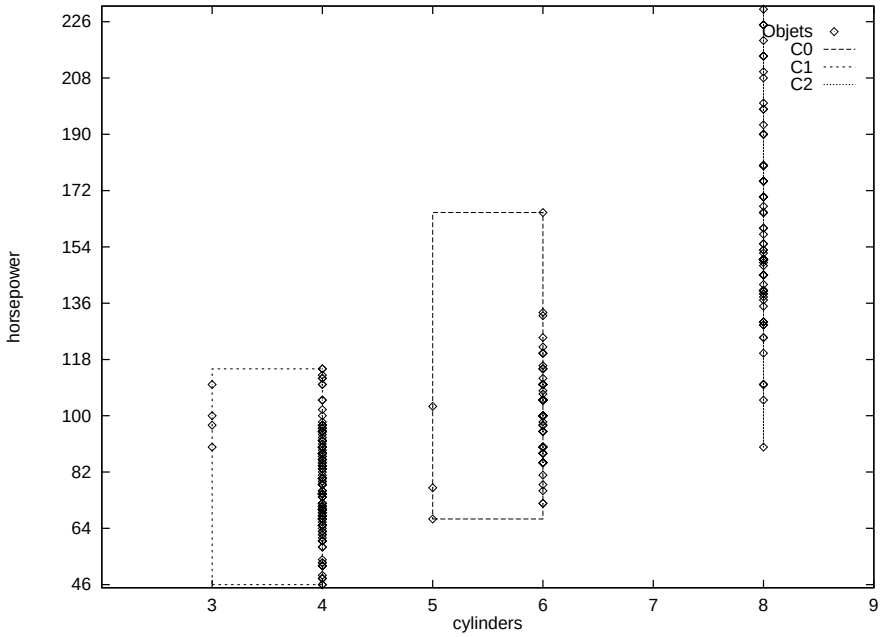
Enfin, nous avons également confronté notre méthode au jeu de données *tic-tac-toe* de l'UCI, qui ne contient que des attributs catégoriels. Ce jeu de données regroupe l'ensemble des fins de partie possibles d'un jeu du morpion. Chaque exemple est donc décrit par 9 attributs catégoriels représentant les 9 cases du plateau du morpion, et prenant chacun 3 modalités possibles : la présence d'une croix, d'un rond, ou bien une case vide.

Si l'on considère les 626 exemples correspondant aux fins de partie gagnantes pour les croix, la difficulté du problème réside dans le fait que les concepts qui régissent la notion de gain d'une partie pour les croix est disjonctif. En effet, il existe 8 façons possibles pour les croix de gagner : 3 croix sur une ligne, pour chacune des 3 lignes, 3 croix sur une colonne, pour chacune des 3 colonnes, et 3 croix en diagonale, pour chacune des 2 diagonales.

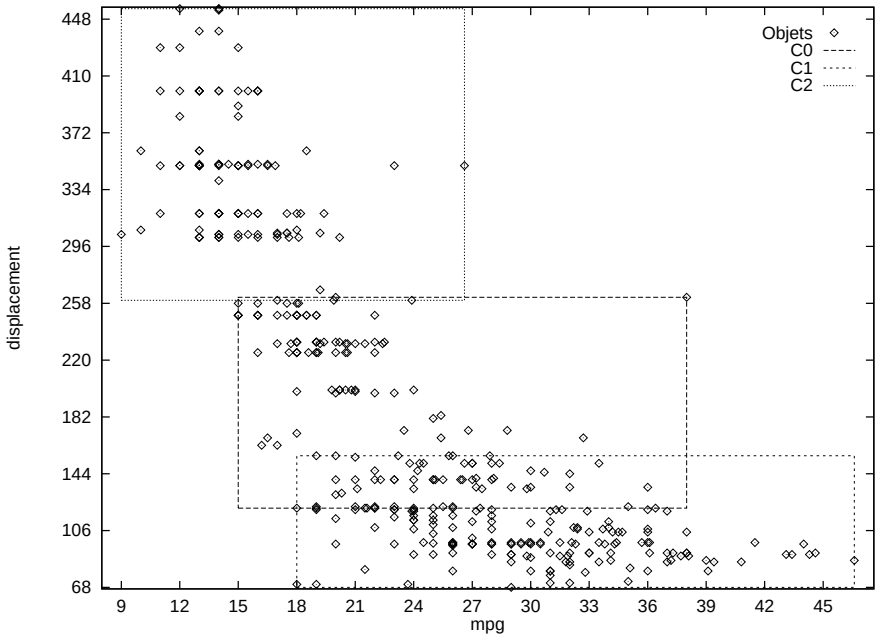
Par ailleurs, ces concepts sont définis dans différents sous-espaces de l'espace original de description des objets, ce qui justifie l'intérêt d'utiliser une méthode de subspace clustering dans ce cadre. En effet, chacun des concepts présentés permettant aux croix de gagner est défini sur un ensemble de 3 attributs qui sont différents pour chacun des concepts. Finalement, **SuSE** exécuté sur ce jeu de données identifie bien la présence de 8 clusters, définis chacun par 3 attributs spécifiques.

7.2.2 Données Pertinence

L'algorithme **SuSE** a donné lieu à un prototype au sein du logiciel de la société Pertinence, et plusieurs expériences ont été menées sur des données réelles de clients de la société, issues pour la plupart de processus industriels complexes. Afin de préserver la confidentialité de ces données, nous ne pouvons entrer dans les détails de ces expériences. Nous présentons néanmoins une partie de nos résultats.

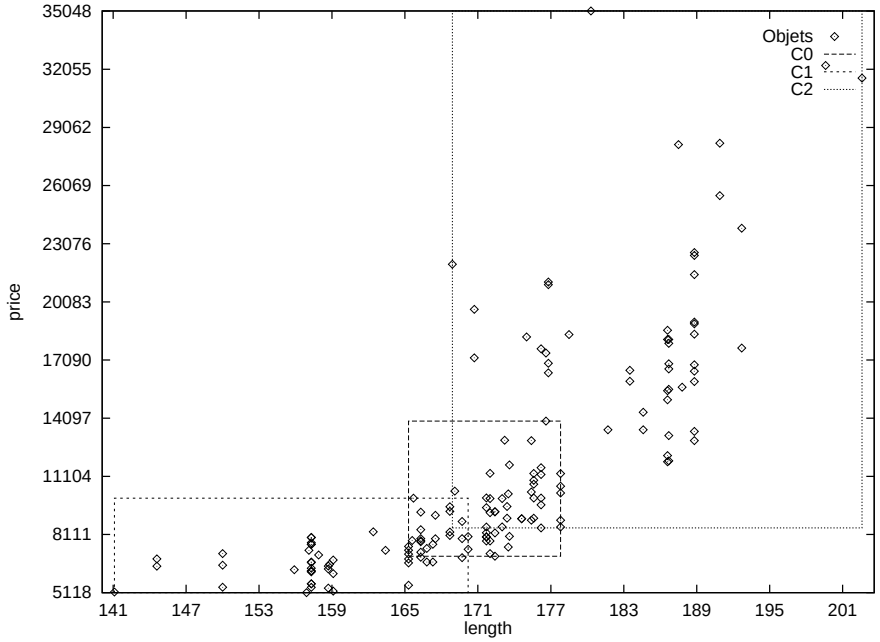


(a) Projections sur *cylinders* et *horsepower*.

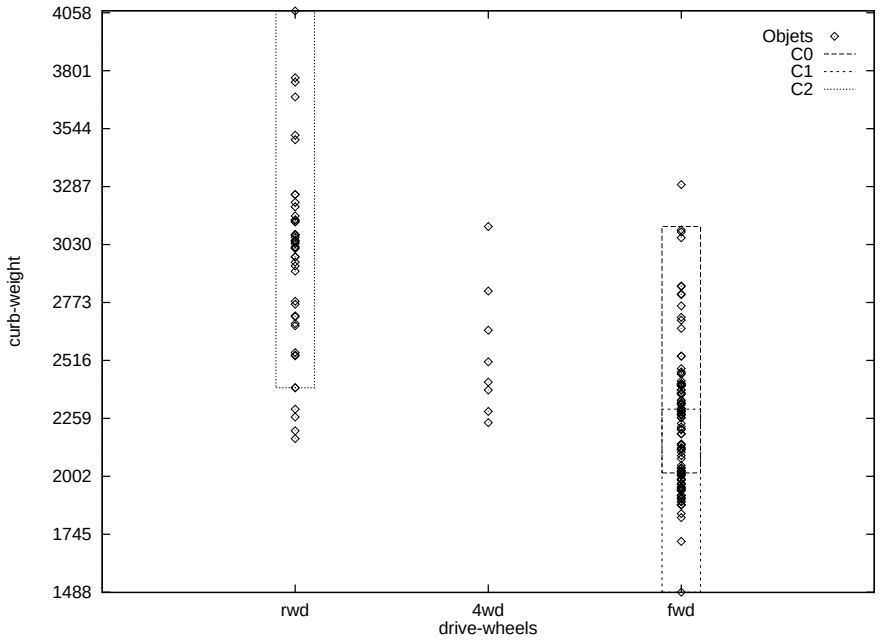


(b) Projections sur *mpg* et *displacement*.

FIG. 7.15 – Visualisation graphique des résultats de **SuSE** sur le jeu de données Auto-Mpg.



(a) Projections sur *longueur* et *prix*.



(b) Projections sur *traction* et *poids*.

FIG. 7.16 – Visualisation graphique des résultats de **SuSE** sur le jeu de données Automobile.

Les jeux de données traités sont étiquetés, et **SuSE** est utilisé en pré-traitement de l'analyse supervisée. Dans certains cas, bien que les attributs de classe sont ignorés lors de l'apprentissage non supervisé, **SuSE** exhibe certaines informations sur les classes, mais nous verrons qu'il peut également arriver qu'il mette en avant certains sous-concepts présents dans les données mais non liés aux classes du problème, informations qui se révèlent alors tout à fait importantes pour appréhender au mieux le problème.

Nous avons choisi de démarrer ces études en recherchant d'abord si les données s'organisaient naturellement en deux groupes. Autrement dit, nous avons positionné le paramètre K de notre algorithme, qui spécifie le nombre de clusters recherchés, à deux. Dans la majorité des cas, cette approche était suffisante pour mettre en avant certaines caractéristiques des jeux de données étudiés, la technique de visualisation graphique des règles obtenues offrant ensuite un complément d'information tout à fait pertinent pour une meilleure compréhension des phénomènes mis en jeu.

La figure 7.17 montre tout d'abord les résultats obtenus sur un jeu de données contenant 67 objets, 32 dimensions numériques, 40 dimensions catégorielles, et 3 classes (bon, moyen et mauvais), que nous nommerons J_1.

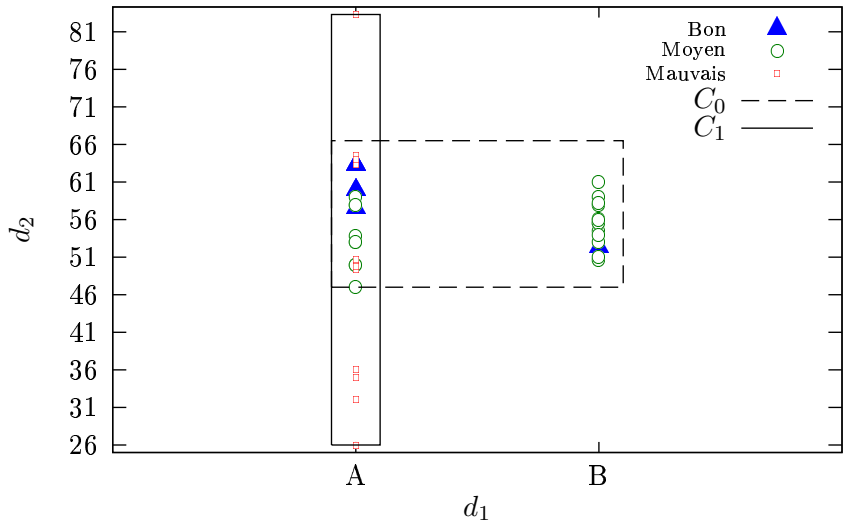


FIG. 7.17 – Visualisation graphique des résultats de **SuSE** sur le jeu de données J_1.

Nous pouvons dès lors observer que la majorité des objets sont définis par des valeurs comprises entre 47 et 67 sur la dimension d_2, et que les objets qui sortent de cet intervalle sont de la classe des *Mauvais*. Les valeurs prises par les objets sur cette dimension correspondant aux mesures de certains capteurs d'un processus industriel complexe, une telle information va alors permettre de repérer l'apparition d'une anomalie dans le processus lorsque la valeur d'un objet sort de cet intervalle [47,67].

Par ailleurs, nous pouvons également remarquer qu'aucun des objets qui sont de la catégorie B sur la dimension d_1 n'est de la classe des *Mauvais*. Dans ce cas, la variable

correspondante du processus n'étant que partiellement contrôlable, cette information permet de mettre l'accent sur l'intérêt de se positionner dans la situation B sur la variable d_1 dès que cela est possible.

Le jeu de données suivant que nous considérons sera nommé J_2. Les données sont issues de la maintenance d'un processus industriel complexe qui consiste à maintenir à une certaine température un liquide contenu dans un récipient. Des capteurs, installés en différents endroits du récipient, mesurent régulièrement sa température. Le jeu de données correspondant contient ainsi 1421 exemples définis par 21 attributs numériques et 3 attributs catégoriels qui représentent les valeurs des 24 capteurs collectées à intervalles réguliers. La figure 7.18 montre une partie des résultats que nous avons obtenus sur ce jeu de données.

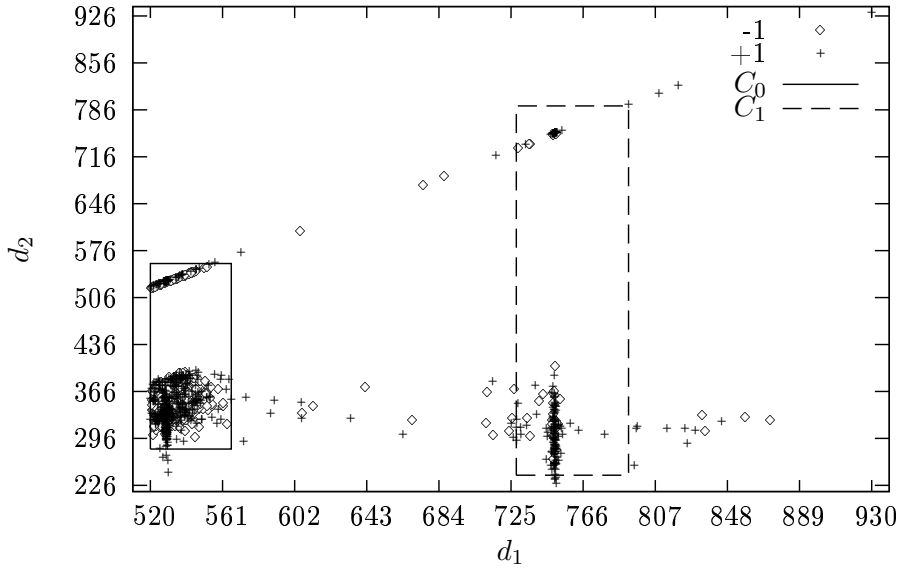


FIG. 7.18 – Visualisation graphique des résultats de **SuSE** sur le jeu de données J_2.

Dans ce cas, deux groupes bien séparés ont été ciblés. On peut d'ailleurs tout de suite remarquer l'intérêt de la détection du bruit présent dans les données, qui permet d'éviter que les quelques objets exceptionnels présents entre les deux groupes ne dégradent les résultats. Ces deux groupes reflètent l'existence de deux classes d'instants particuliers dans le processus de production, qui correspondent au fait qu'à intervalles réguliers, le liquide du récipient est changé, et la température augmentée momentanément. Les phénomènes mis en jeu à ces différents instants étant différents, et par conséquent les conditions de production également, cette information a finalement justifié la tenue de deux études séparées.

La visualisation graphique permet également d'observer que dans chacun des deux groupes identifiés, deux sous-groupes sont présents. Après une observation plus approfondie des données par les experts, il s'est avéré que ce phénomène était dû à une période

pendant laquelle les valeurs de deux capteurs avaient été inversées. Cette deuxième information a donc permis de mettre à jour les valeurs de ces capteurs afin d'éliminer ce bruit qui était présent dans les données.

La dernière étude que nous présentons concerne un jeu de données que nous nommerons J_3 et qui contient 142 objets définis par 16 dimensions numériques, et séparées en 3 classes : bon, moyen et mauvais. La figure 7.19 montre les premiers résultats que nous avons obtenus sur ce jeu de données.

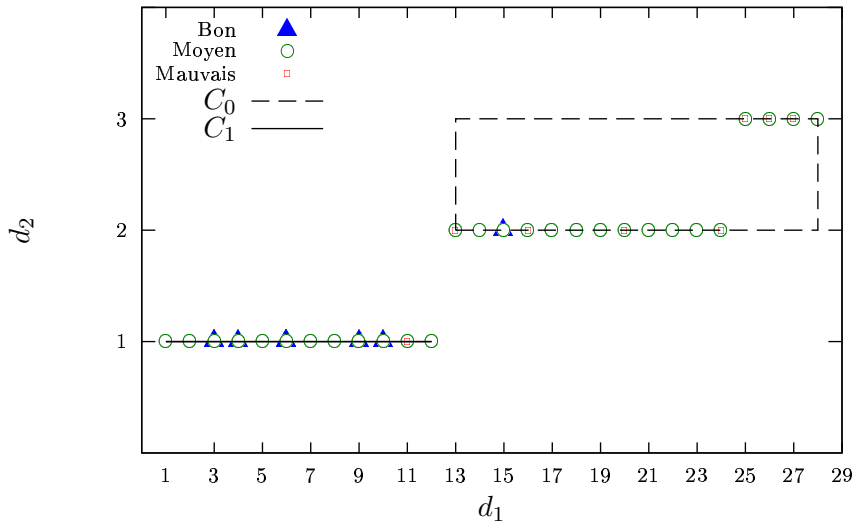


FIG. 7.19 – Visualisation graphique des résultats initiaux de **SuSE** sur le jeu de données J_3.

Ces premiers résultats mettent en avant l'existence d'une corrélation entre les deux dimensions d_1 et d_2 sélectionnées par l'algorithme, justifiant par là même la suppression de la dimension d_2 pour mener une deuxième étude. La figure 7.20 présente les résultats obtenus lors de cette deuxième étude.

Sur la première projection de la figure 7.20, on observe une seconde corrélation entre les deux dimensions d_0 et d_1, ainsi que la présence d'un objet bruité pour $d_0 = 56$ et $d_1 = 24$. On remarque donc à ce niveau que, malgré l'utilisation de l'hypothèse d'indépendance des dimensions sur laquelle est basée notre méthode de clustering, la méthode de visualisation graphique des résultats que nous avons proposée permet de mettre en avant l'existence de telles corrélations entre dimensions, ainsi que la présence de bruit dans les données.

Sur la seconde projection de la figure 7.20, on observe une augmentation des valeurs des objets sur la dimension d_3 lorsque les valeurs sur la dimension d_1 sont supérieures à 16. On observe également la cohérence des deux groupes ciblés par **SuSE**, qui reflètent deux phénomènes différents. On peut enfin remarquer sur cette projection la présence de davantage d'objets classés *Mauvais* lorsque leurs valeurs sur d_1 sont supérieures à 16, alors qu'au contraire, l'ensemble des objets classés *Bon* sont définis par de plus faibles

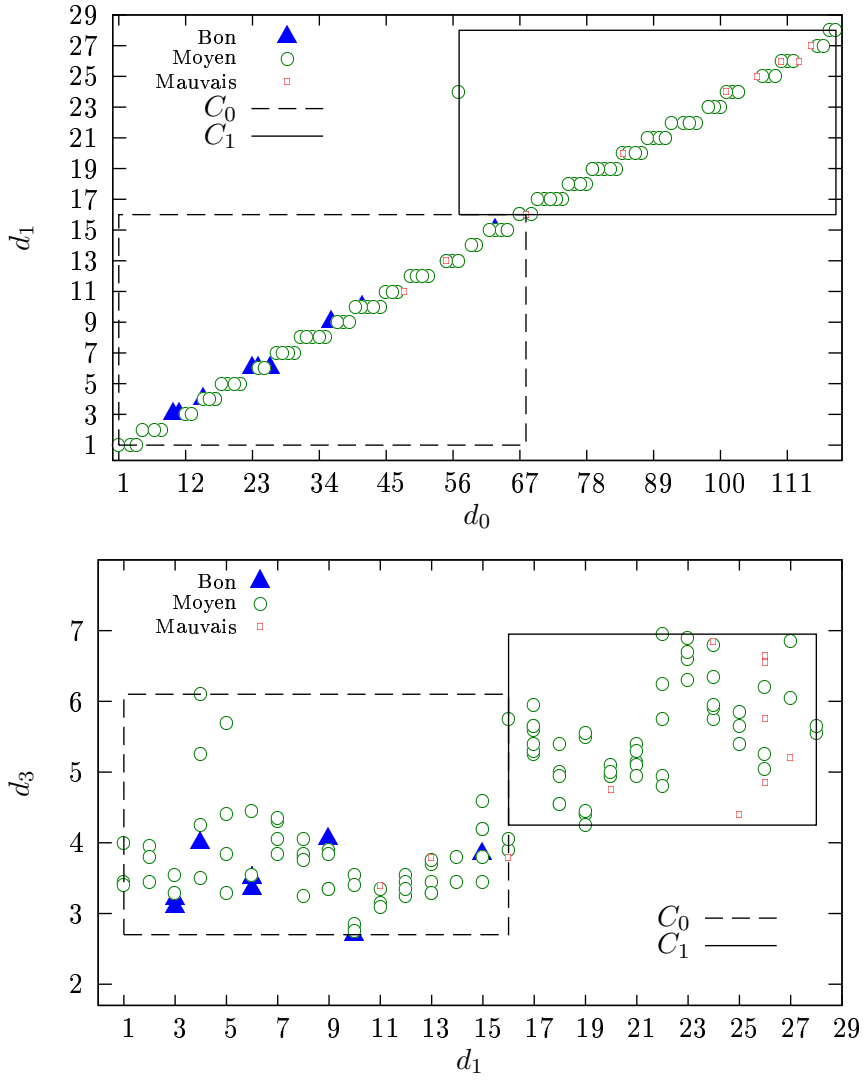


FIG. 7.20 – Visualisation graphique des résultats de **SuSE** sur le jeu de données J_3.

valeurs sur d_1 ainsi que sur d_3.

Finalement, on peut donc remarquer que **SuSE** et sa méthode de visualisation graphique des résultats permettent de mettre en avant différents types d'informations dans les données. Tout d'abord, la présence de corrélations entre dimensions ou d'objets bruités est facilement observable sur des projections à deux dimensions. Ensuite, bien que n'utilisant aucune information de classe, **SuSE** peut exhiber certaines informations sur ces classes. Enfin, lorsque différents sous-concepts non liés aux classes sont présents dans les données, alors leur détection par **SuSE** justifie souvent la tenue d'études séparées sur ces différents sous-ensembles d'objets détectés.

7.2.3 Données semi-structurées

Dans le cadre du challenge INEX traitant de la classification de documents XML à partir de leur structure uniquement, nous avons également mené plusieurs expérimentations. Nous avons ainsi pu confronter notre méthode à des jeux de données contenant de nombreux objets définis par de nombreuses dimensions, nous permettant ainsi d'observer le comportement de notre méthode à grande échelle. Les méthodes mises en œuvre dans ce cadre sont présentées dans la section 6.2.

Nous présentons tout d'abord les résultats obtenus par une méthode de classification supervisée basée sur la transformation des documents XML sous forme d'attributs-valeurs, et sur l'ensemble des jeux de données proposés : *inex-s* correspond à un jeu de données réel d'articles scientifiques organisés selon différents formats ; *m-db-s-0* correspond à un jeu de données créé à partir de bases de données réelles concernant les descriptions de nombreux films, répartis en 11 groupes dépendant de la structure utilisée pour présenter ces films ; les jeux de données *m-db-s-1*, *m-db-s-2* et *m-db-s-3* correspondent à des versions de plus en plus bruitées du jeu de données *m-db-s-0*. Le tableau 7.3 montre le nombre d'objets utilisés en apprentissage et en test pour chacun de ces jeux de données.

jeu de données	nombre d'objets d'apprentissage	nombre d'objets test
inex-s	6057	6053
m-db-s-0	3866	4816
m-db-s-1	4846	4815
m-db-s-2	4845	4816
m-db-s-3	4843	4818

TAB. 7.3 – Nombre d'objets des jeux de données de documents XML.

Nous présentons ensuite les résultats obtenus en clustering par l'adaptation de **SuSE** au cas des données XML, ainsi que les résultats obtenus en classification supervisée par l'adaptation de **SuSE** au cas des données XML dans le cadre supervisé. En particulier, nous montrons l'efficacité de ces méthodes dans l'objectif de présenter les résultats obtenus dans un format compréhensible par l'utilisateur.

En classification supervisée, un ensemble de données étiquetées servant d'ensemble d'apprentissage est d'abord fourni pour la construction du modèle, puis des données non étiquetées servant d'ensemble de test sont fournies pour évaluer le modèle. Les différentes mesures utilisées pour évaluer les résultats sont alors les *micro-rappel* et *macro-rappel*, présentées dans [Denoyer et al., 2006], qui doivent être maximisées pour optimiser les résultats.

En classification non supervisée, seules des données sans étiquette sont fournies. Les clusters ciblés étant connus, des mesures de la qualité externe du clustering proposé sont alors utilisées pour évaluer les résultats. Ces mesures, présentées dans [Denoyer et al., 2006], sont la *micro pureté*, la *macro pureté*, la *micro entropie*, la *macro entropie* et l'*information mutuelle*. Les deux premières mesures ainsi que la cinquième doivent être maximisées pour optimiser la correspondance entre les résultats produits par le clustering et les résultats attendus. À l'inverse, les mesures d'entropie doivent être minimisées pour optimiser les résultats obtenus.

Tout d'abord, le tableau 7.4 présente le nombre de nouveaux attributs générés lors de la transformation (telle que décrite dans la section 6.2) des arbres XML en attributs-valeurs pour chaque jeu de données. Nous pouvons dès lors observer que de très grandes quantités de nouveaux attributs sont générées par cette méthode. En particulier, le nombre d'attributs représentant l'ensemble des chemins dans les arbres est très important.

jeu de données	nombre de labels	nombre de relations père-fils	nombre de relations frères	nombre de positions des nœuds	nombre de chemins	total
inex-s	150	1038	827	2475	3674	8164
m-db-s-0	197	2172	419	6575	320	9683
m-db-s-1	197	6477	5617	9159	16772	38222
m-db-s-2	196	8953	7455	9183	25628	51415
m-db-s-3	199	10639	9557	8537	37576	66508

TAB. 7.4 – Nombre d'attributs générés pour chaque jeu de données XML.

Nous avons exécuté l'algorithme C5 [Quinlan, 2004] boosté 10 fois sur ces jeux de données. Cependant, comme le nombre d'attributs s'est révélé trop important pour C5 dans le cas des jeux de données *m-db-s-2* et *m-db-s-3*, nous avons exclu les attributs représentant les chemins dans les arbres pour ces jeux de données. Les taux d'erreur obtenus en exécutant dix validations croisées sur l'ensemble des jeux de données sont présentés dans le tableau 7.5 qui montre que cette méthode obtient des taux d'erreur tout à fait raisonnables, motivant par là même l'intérêt d'utiliser de telles transformations d'arbres XML en attributs-valeurs.

Le tableau 7.6 reporte ensuite les *micro-rappel* et *macro-rappel* calculés sur les jeux de test fournis lors du challenge. Les résultats montrent donc la robustesse de notre méthode en présence de bruit dans les données, puisque les taux d'erreur restent stables pour l'ensemble des jeux de données *m-db-s*. Ces résultats sont les meilleurs obtenus

jeu de données	taux d'erreur
inex-s	1.1%
m-db-s-0	2.6%
m-db-s-1	3.8%
m-db-s-2	6.2%
m-db-s-3	6.2%

TAB. 7.5 – Taux d'erreur de C5 boosté sur les jeux de données XML transformés en attributs-valeurs.

lors du challenge, réunissant neuf participants pour cette tâche.

jeu de données	micro-rappel	macro-rappel
inex-s	0.941	0.958
m-db-s-0	0.968	0.960
m-db-s-1	0.966	0.956
m-db-s-2	0.942	0.932
m-db-s-3	0.947	0.935

TAB. 7.6 – Micro-rappel et macro-rappel de C5 boosté sur les jeux de tests XML transformés en attributs-valeurs.

En ce qui concerne la méthode de clustering que nous avons adaptée pour faire face au cas des données XML, présentée dans la section 6.2, nous avons observé que, pour le jeu de données *m-db-s-0* par exemple, la méthode a correctement identifié les classes numérotées de 1 à 5 ; elle a mélangé les classes 7 et 9 ; et les classes restantes ont également été mélangées. Les différentes mesures calculées lors du challenge ont alors mené à une *micro pureté* de 0.78, une *macro pureté* de 0.75, une *micro entropie* de 0.18, une *macro entropie* de 0.21, et une *information mutuelle* de 1.87 sur les données de test. Ces résultats faisaient également partie des meilleurs, classant la méthode deuxième sur six participants pour cette tâche.

Par ailleurs, la hiérarchie formée, présentée par la figure 7.21, met en avant la capacité de notre méthode à fournir un résultat interprétable. $R4$, $R6$, $R7$ et $R10$ correspondent à des règles définies sur 10 attributs :

- $R4$ est définie sur des attributs représentant les relations frères dans les arbres ;
- $R6$ est définie sur des attributs représentant les labels des arbres ;
- $R7$ et $R10$ concernent le nombre de fils associés à différentes positions de nœuds dans les arbres.

Enfin $R11$ correspond à tester si le nombre de chemins (BE-AL-AT) dans les arbres est inférieur ou égal à 1, et s'il n'y a aucun chemin (BE-AL-AT-AR).

L'adaptation de notre méthode à l'apprentissage supervisé a également mené à des résultats très intéressants. L'arbre de décision obtenu dans ce cas est présenté par la figure 7.22. Nous observons alors que le résultat produit est tout à fait compréhensible.

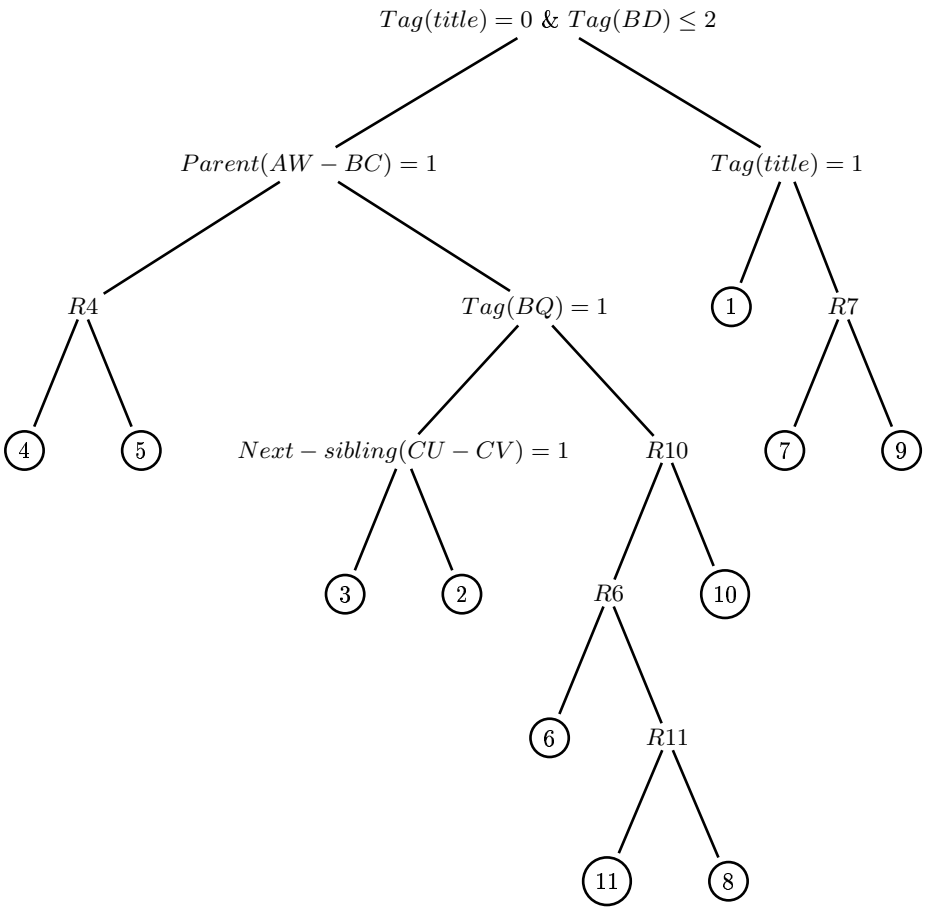


FIG. 7.21 – Arbre de décision obtenu par le clustering du jeu de données m-db-s-0.

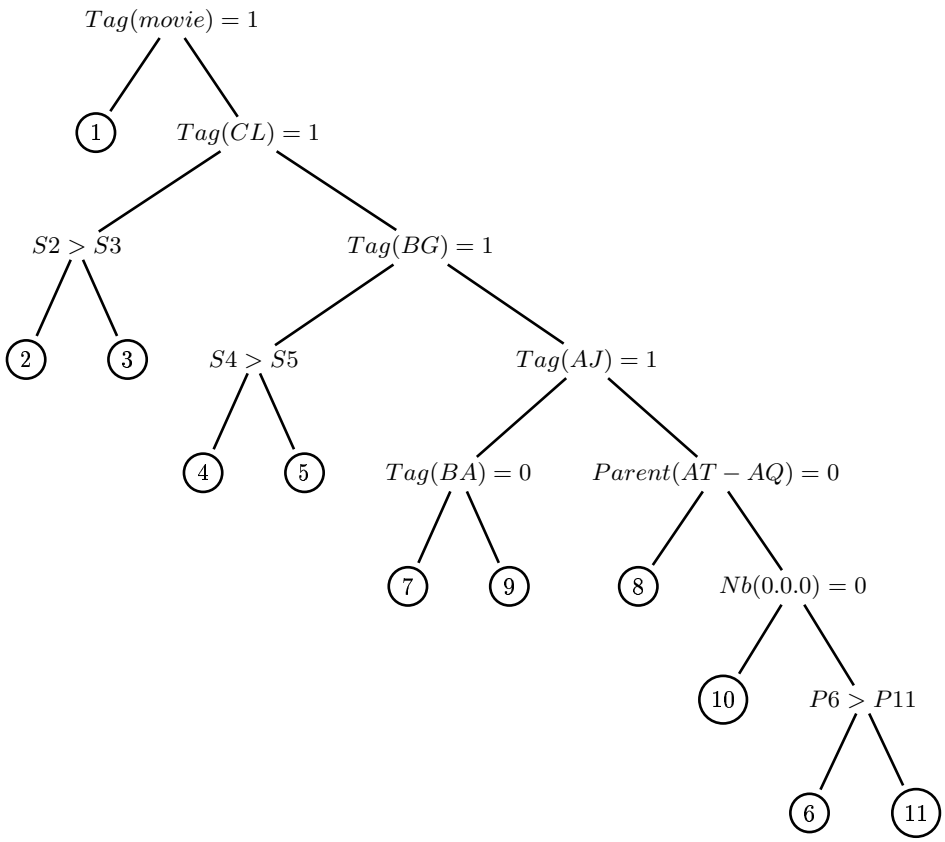


FIG. 7.22 – Arbre de décision obtenu en classification supervisée sur le jeu de données m-db-s-0.

- $S2$, $S3$, $S4$ et $S5$ représentent des tests probabilistes sur des modèles basés sur les relations frères des arbres, concernant respectivement les classes 2, 3, 4 et 5 ;
- $P6$ et $P11$ représentent des tests probabilistes sur des modèles basés sur les chemins dans les arbres, concernant les classes 6 et 11 ;
- et $Nb(0.0.0)$ indique le nombre de fils du premier petit-fils de la racine de l'arbre.

Ainsi, par exemple, l'appartenance à la classe 1 ne dépend que de la présence du label nommé *movie*. Et de la même façon, l'appartenance à la classe 8 ne dépend que de l'absence des labels *movie*, CL, BJ, AJ, et de la présence d'une relation père-fils entre les labels AT et AQ.

Le taux d'erreur d'un tel arbre de décision en validation croisée sur l'ensemble d'apprentissage du challenge est de 3%, ce qui correspond à très peu d'erreurs de classification à l'exception de plusieurs mélanges entre les classes 6 et 11. Sur les données de test du challenge, les résultats de cette méthode sont aussi tout à fait intéressants, avec un *micro-rappel* de 0.906 et un *macro-rappel* de 0.924, menant à classer la méthode cinquième sur neuf participants, mais celle-ci se distinguant des autres par sa capacité à présenter un résultat interprétable à l'utilisateur.

7.3 Bilan

Dans la première partie de ces expérimentations sur des données artificielles, nous avons pu observer le bon comportement des méthodes de subspace clustering que nous avons proposées, **Tuareg** étant tout à fait efficace dans certains cas d'application particuliers où l'hypothèse de séparabilité des clusters sur au moins une dimension de description est respectée, alors que **SuSE** est capable d'identifier les clusters présents dans les données même quand cette hypothèse n'est pas vérifiée, mais nécessite pour cela un temps d'exécution plus important.

Dans la seconde partie de ces expérimentations portant sur différents types de données réelles, nous avons pu observer la pertinence des résultats fournis par **SuSE**, ainsi que l'intérêt de la méthode de visualisation graphique des résultats que nous avons proposée. Nous avons également pu constater sa capacité à faire face à de larges bases de données bruitées et à fournir en sortie des résultats facilement interprétables par l'utilisateur.

Ces bons résultats ne permettent cependant pas de généraliser l'intérêt de **SuSE** à n'importe quel cas d'application. Ces expériences ne permettent pas non plus de mettre en avant sa supériorité face à d'autres méthodes de clustering existantes. Face à cette difficulté dans l'évaluation de l'intérêt des méthodes de clustering, nous consacrons la partie suivante à cette problématique, et proposons une nouvelle méthode d'évaluation plus globale, objective et quantitative que les méthodes existantes, et montrons expérimentalement son intérêt.

Troisième partie

Nouvelle méthode d'évaluation d'algorithmes de clustering

Chapitre 8

Évaluation en cascade

Ce chapitre se place dans le cadre de l'évaluation des résultats d'algorithmes de clustering, et de la comparaison de tels algorithmes. Dans la section 2.5, nous avons présenté les différentes techniques existantes pour effectuer de telles évaluations, et nous avons mis en avant leurs limitations.

La critique principale que l'on peut faire à ces techniques est que les résultats obtenus ne peuvent être généralisés à différents cas d'application. En effet, que l'évaluation soit effectuée à partir de données artificielles, ou à partir de données réelles et à l'aide d'un expert, les résultats obtenus ne peuvent être généralisés à différents types de données réelles. De plus, dans le cas de données réelles, s'il est possible pour un expert d'indiquer si un résultat donné a du sens, il est beaucoup plus problématique de préciser si un résultat est meilleur qu'un autre, ce qui empêche la comparaison entre différentes méthodes de clustering. Enfin, aucun critère de la qualité interne d'un clustering ne peut être considéré comme pertinent dans n'importe quel cas d'application car les propriétés des données et les critères d'intérêt du clustering peuvent varier d'une application à l'autre.

Nous proposons donc dans ce chapitre une nouvelle méthode d'évaluation et de comparaison des résultats d'algorithmes de clustering que nous appelons *évaluation en cascade* [Candillier et al., 2006a], et nous justifions son intérêt, en montrant en particulier que, contrairement aux méthodes existantes, l'évaluation dans ce cadre est alors plus globale, objective, et quantitative.

8.1 Présentation générale

Le risque principal dans l'évaluation d'une méthode de clustering est de considérer celle-ci comme un but en soi. En fait, ce que l'on cherche à évaluer est la capacité d'une méthode de clustering à identifier de l'information nouvelle, intéressante et utile, c'est-à-dire une connaissance nouvelle qu'il est intéressant d'utiliser dans un certain cadre. Nous attendons également que la méthode soit capable d'identifier de telles informations intéressantes sur différents types de problèmes.

L'idée principale de notre approche est donc de considérer le clustering comme un

prétraitement à une autre tâche que nous sommes capables d'évaluer : l'apprentissage supervisé par exemple. La nouvelle méthode d'évaluation que nous proposons consiste à comparer les résultats d'un algorithme supervisé lorsqu'il est (ou pas) aidé par de l'information issue d'un algorithme de clustering. Ainsi, si les résultats de l'algorithme supervisé sont améliorés lorsqu'il utilise de l'information issue d'un algorithme de clustering, alors nous estimons que cela signifie que cette information est nouvelle et utile.

Cette méthode nous permet donc d'évaluer objectivement l'intérêt de l'information apportée par l'algorithme de clustering. De plus, la diminution de l'erreur de l'algorithme supervisé aidé par l'information issue des résultats du clustering nous permet également de quantifier cet intérêt.

Notre méthode se place dans le cadre de la combinaison de classifieurs, celle d'une méthode supervisée et d'une méthode non supervisée dans notre cas. Plusieurs façons de combiner différents classifieurs par des votes peuvent être trouvées dans [Ali and Pazzani, 1996], les deux méthodes de vote les plus reconnues étant le *bagging* [Breiman, 1996a] et le *boosting* [Freund and Schapire, 1996]. Quelques généralisations théoriques de ces techniques ont également été étudiées, menant aux *arcing classifiers* [Breiman, 1996b], aux *méthodes d'ensemble* [Dietterich, 2000] et aux *leveraging methods* [Meir and Rätsch, 2003].

Nous nous focalisons ici sur les techniques qui utilisent différents apprenants de manière séquentielle. Dans de telles méthodes, la sortie d'un apprenant est un enrichissement de la description des exemples qui est ensuite utilisé par l'apprenant suivant. Dans ce cadre, la *stacked generalization* [Wolpert, 1992] est une méthode très générale dans laquelle différents traitements sont enchaînés : chaque traitement modifie la représentation des exemples, et le nouveau jeu de données enrichi est utilisé par le niveau suivant.

La *cascade generalization* [Gama and Brazdil, 2000] est un cas particulier de stacked generalization : à chaque niveau, un classifieur est appliqué sur chaque exemple $\overrightarrow{x_i}$, fournissant la probabilité $p(c|\overrightarrow{x_i})$ que $\overrightarrow{x_i}$ appartienne à la classe c ; ces probabilités sont ensuite ajoutées à la description des exemples et utilisées par le classifieur suivant. La cascade generalization permet de combiner plusieurs classifieurs mais en pratique, seulement deux classifieurs sont utilisés.

Enfin, nous considérons également le cas où un apprenant supervisé et un apprenant non supervisé sont combinés comme proposé dans [Apte et al., 2002]. Dans ce cas, plusieurs clusterings sont exécutés en faisant varier les paramètres d'entrée de la méthode (par exemple le nombre de groupes recherchés), menant ainsi à différentes partitions de l'ensemble des objets. Pour chaque partition, plusieurs apprentissages supervisés sont menés indépendamment sur les différents groupes d'objets formés. Et finalement, la partition ayant mené au taux d'erreur le plus faible est conservée.

Basée sur ce principe d'enchaîner en cascade un apprenant non supervisé et un apprenant supervisé, puis de calculer la diminution du taux d'erreur de l'apprenant supervisé lorsqu'il est aidé par l'information issue de l'apprenant non supervisé, la nouvelle méthode d'évaluation d'algorithmes de clustering que nous proposons s'appelle *évaluation en cascade*.

8.2 Nouvelle méthodologie d'évaluation

Étant donné un jeu de données initial contenant des informations de classes, les étapes principales de la méthode que nous proposons sont les suivantes :

1. apprentissage 1 :
   - apprentissage supervisé sur le jeu de données initial.
2. apprentissage 2 :
   - clustering sur le jeu de données en ignorant les informations de classes ;
   - enrichissement du jeu de données à partir des résultats du clustering ;
   - apprentissage supervisé sur le jeu de données enrichi.
3. comparaison des résultats des deux classifieurs appris.

Comme nous l'avons évoqué précédemment, nous envisageons deux façons d'enrichir les jeux de données à partir des résultats du clustering. La première consiste à créer de nouveaux attributs représentant l'information capturée par le clustering, et à ajouter ces nouveaux attributs dans le jeu de données initial. La seconde est de créer des sous-jeux de données en fonction des groupes formés par le clustering.

Concernant les nouveaux attributs créés depuis les résultats du clustering dans le cas de la première méthode, différents types d'informations peuvent être ajoutés.

1. Étant donné que la plupart des algorithmes de clustering fournissent en sortie une partition de l'ensemble des objets, nous pouvons utiliser comme nouveaux attributs l'appartenance des objets aux clusters. Cette information serait ainsi représentée par un nouvel attribut catégoriel, chaque objet étant associé à un identifiant du cluster auquel il appartient.
2. On peut également associer à chaque objet un ensemble d'attributs représentant le centre du cluster auquel il est associé. Le nombre d'attributs du jeu de données serait ainsi doublé.
3. Dans le cas du subspace clustering, nous pouvons également associer à chaque objet un nouvel attribut numérique par attribut initial correspondant au poids de cet attribut pour le cluster auquel l'objet appartient. Un tel nouvel attribut permettrait de différencier les objets pour lesquels un attribut donné est pertinent des autres objets pour lesquels cet attribut n'est pas pertinent (selon les résultats du subspace clustering).
4. La probabilité d'appartenance de chaque objet à chaque cluster peut également constituer une nouvelle information à ajouter lorsque des modèles probabilistes sont utilisés. Si ce n'est pas le cas, alors la distance de chaque objet à chaque cluster peut être calculée puis ajoutée à leur description.
5. Les valeurs minimum et maximum des coordonnées des membres de chaque cluster sur chaque attribut peuvent aussi être utilisées pour enrichir les jeux de données. Autrement dit, à chaque objet serait alors associé l'ensemble des valeurs définissant le *moindre généralisé* du cluster auquel il appartient, et le nombre d'attributs du jeu de données serait ainsi triplé.

Par ailleurs, comme la plupart des algorithmes de clustering nécessitent de spécifier différents paramètres en entrée, nous pouvons exécuter ces algorithmes pour différentes valeurs de paramètres, et enrichir les jeux de données pour chaque résultat de clustering. En particulier, de nombreuses méthodes de clustering nécessitent de spécifier le nombre de clusters recherchés. Dans ce cas, nous pouvons exécuter la méthode plusieurs fois en faisant varier le nombre de clusters entre 2 et 10 par exemple, et cumuler tous les attributs produits dans la nouvelle description des exemples. L'algorithme supervisé utilisé ensuite pourrait alors choisir quel(s) attribut(s) utiliser parmi tous ceux produits.

Dans le cas de la seconde méthode de combinaison proposée, nous générons dans un premier temps plusieurs partitions avec différents paramètres d'entrée. Puis nous calculons les erreurs en validation croisée d'apprenants supervisés exécutés indépendamment sur chaque groupe d'objet créé par le clustering. Et finalement, la partition ayant mené au taux d'erreur le plus faible est conservée.

La figure 8.1 résume les étapes principales de cette méthodologie :

1. diviser le jeu de données en deux parts égales : un ensemble d'apprentissage Ap^- et un ensemble de test Te^- ;
2. exécuter plusieurs fois le clustering, avec différents paramètres d'entrée et en ignorant les informations de classes, et produire les ensembles d'apprentissage Ap^+ et de test Te^+, enrichissements des ensembles Ap^- et Te^- ;
3. lancer l'apprentissage supervisé sur l'ensemble d'apprentissage Ap^- et produire le classifieur C^- ;
4. lancer l'apprentissage supervisé sur l'ensemble d'apprentissage enrichi Ap^+ et produire le classifieur C^+ ;
5. lancer la classification sur l'ensemble test Te^- à partir du classifieur C^- ;
6. lancer la classification sur l'ensemble test Te^+ à partir du classifieur C^+ ;
7. et comparer les erreurs de classification.

Pour évaluer l'amélioration des résultats avec ou sans les nouvelles informations fournies par un algorithme de clustering évalué, nous testons les deux méthodes (l'apprentissage supervisé sur les données initiales et l'apprentissage supervisé sur les données enrichies) sur différents jeux de données indépendants. Sur chaque jeu de données, nous faisons cinq validations croisées avec découpage du jeu de données en deux, comme proposé dans [Dietterich, 1998]. Pour chaque validation croisée, nous calculons les taux d'erreur pondérés des deux méthodes. Puis nous utilisons quatre mesures pour comparer les résultats :

1. *nb vict* : le nombre de victoires de chaque méthode ;
2. *vict sign* : le nombre de victoires significatives, en utilisant le *5×2cv F-test* [Alpaydin, 1999] pour vérifier si les résultats sont significativement différents ;
3. *wilcoxon* : le *wilcoxon signed rank test*, qui indique si une méthode est significativement meilleure qu'une autre sur un ensemble de problèmes indépendants ;
4. et *moyenne* : l'erreur pondérée moyenne.

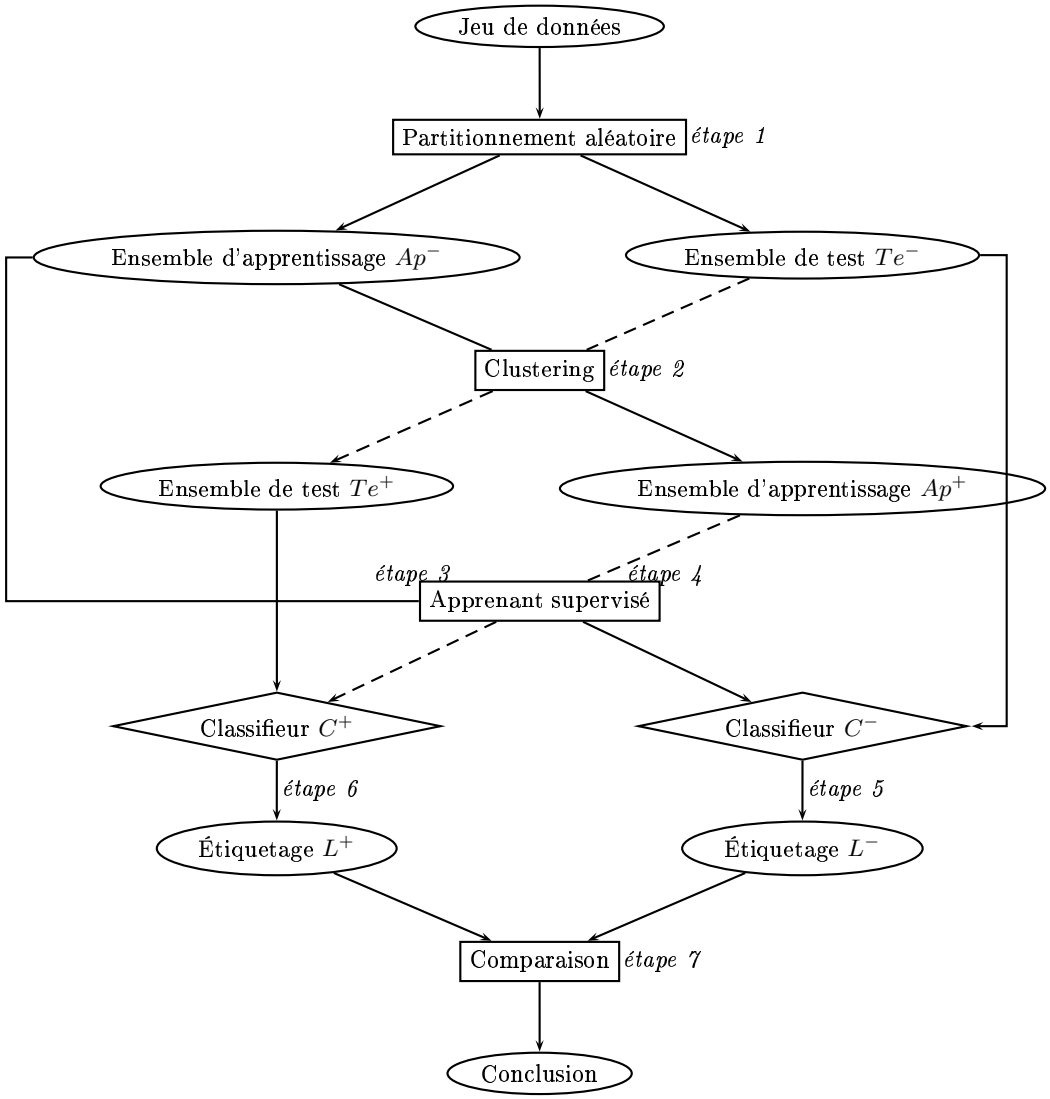


FIG. 8.1 – Méthodologie d'évaluation de l'intérêt d'ajouter à un jeu de données de l'information provenant d'un algorithme de clustering.

Pour effectuer ces comparaisons, C4.5 [Quinlan, 1993] comme apprenant supervisé est bien adapté car, comme il a tendance à utiliser un minimum d'attributs pendant l'apprentissage et fournit en sortie un arbre de décision où l'on peut observer quels attributs ont été utilisés et à quels niveaux ils ont été utilisés, il met clairement en avant si les nouvelles informations issues du clustering sont utiles ou non. De plus, il a également l'avantage d'être rapide et d'être capable de considérer des attributs catégoriels aussi bien que numériques. Dans un premier temps, c'est donc cet algorithme supervisé qui sera utilisé, mais dans la suite, d'autres seront considérés afin d'observer si les résultats de nos évaluations sont dépendants de l'algorithme supervisé utilisé.

8.3 Exemple d'utilisation

Le jeu de données *wine* issu de l'UCI Machine Learning Repository [Blake and Merz, 1998] contient la description de 178 vins définis par 13 attributs numériques. Trois types de vins, qui représentent les classes, sont présents dans la base.

Lorsque C4.5 est appliqué sur ce jeu de données, l'arbre de décision obtenu est celui présenté par la figure 8.2. On observe alors que les attributs sélectionnés par C4.5 sont *ColorIntensity*, *Flavanoids* et *Proline*.

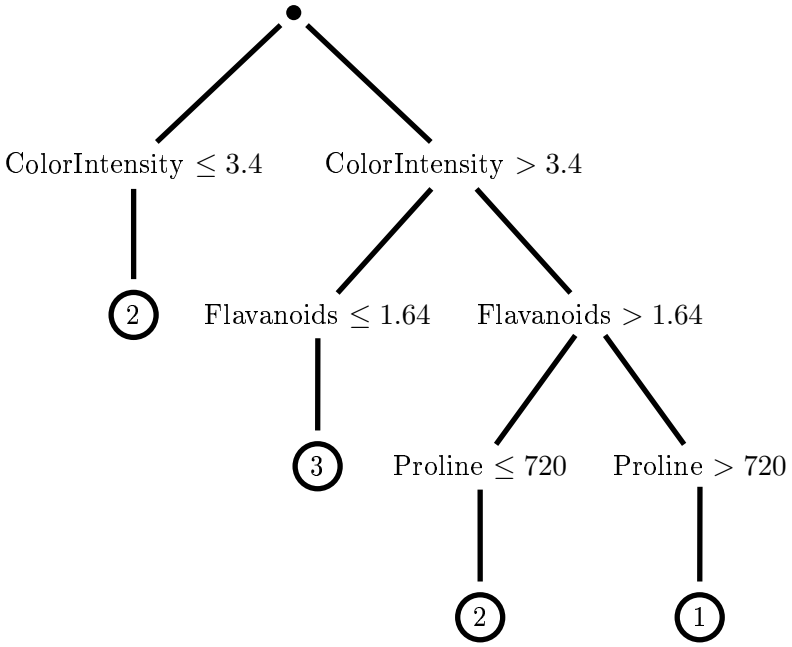


FIG. 8.2 – Arbre de décision appris par C4.5 sur le jeu de données wine.

Lorsque l'on utilise **SuSE** pour ajouter de l'information au jeu de données, nous obtenons l'arbre de décision présenté par la figure 8.3. On observe alors que C4.5 a utilisé tout en haut de l'arbre de décision qu'il a construit l'un des attributs créés par notre méthode : celui qui se réferre à l'appartenance des objets aux clusters lorsque le nombre

de clusters est positioné à 3. L'autre attribut utilisé est *ColorIntensity*, également utilisé par C4.5 sur le jeu de données initial.

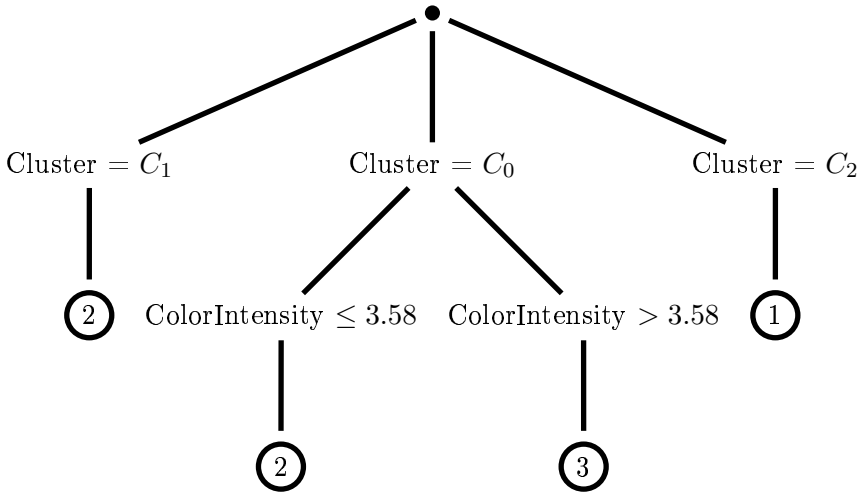


FIG. 8.3 – Arbre de décision appris par C4.5, après ajout d'information par **SuSE**, sur le jeu de données wine.

Si l'on reprend la méthode de visualisation graphique des règles associées aux clusters présentée dans la section 5.7, on s'aperçoit que la meilleure projection en deux dimensions sélectionnée concerne les attributs *Flavanoids* et *Proline*, qui correspondent aux deux autres attributs utilisés dans l'arbre de décision appris par C4.5 initialement. La figure 8.4 montre cette projection.

Finalement, nous observons donc sur cet exemple une correspondance forte entre l'arbre de décision appris par C4.5 sur le jeu de données wine initial, et l'arbre de décision appris par C4.5 sur le jeu de données wine enrichi par les résultats du clustering appris par **SuSE**.

De plus, nous pouvons remarquer que l'attribut ajouté par notre méthode est utilisé par C4.5 dès le début de la construction de l'arbre de décision. Nous sommes donc là en présence d'un exemple où C4.5 a utilisé l'information provenant des résultats du clustering à un haut niveau dans l'arbre de décision qu'il construit. Il semble donc qu'il spécialise ici son traitement selon les différentes zones créées par **SuSE** dans l'espace de description des objets.

Une autre interprétation possible de ce résultat est que le nouvel attribut créé aide C4.5 à définir des zones de décision plus complexes. En effet, utiliser le nouvel attribut ajouté par **SuSE** correspond en fait à effectuer un test sur plusieurs attributs simultanément, alors que C4.5 seul ne peut utiliser à un nœud de l'arbre qu'un attribut à la fois, et que cela peut être difficile pour lui de trouver l'enchaînement de tests équivalent. Notons d'ailleurs à ce niveau que comme **SuSE** est capable de caractériser l'appartenance d'un objet à un cluster de manière compréhensible à l'aide de règles définies par un minimum d'attributs pertinents, le combiner avec C4.5 reste tout à fait approprié à

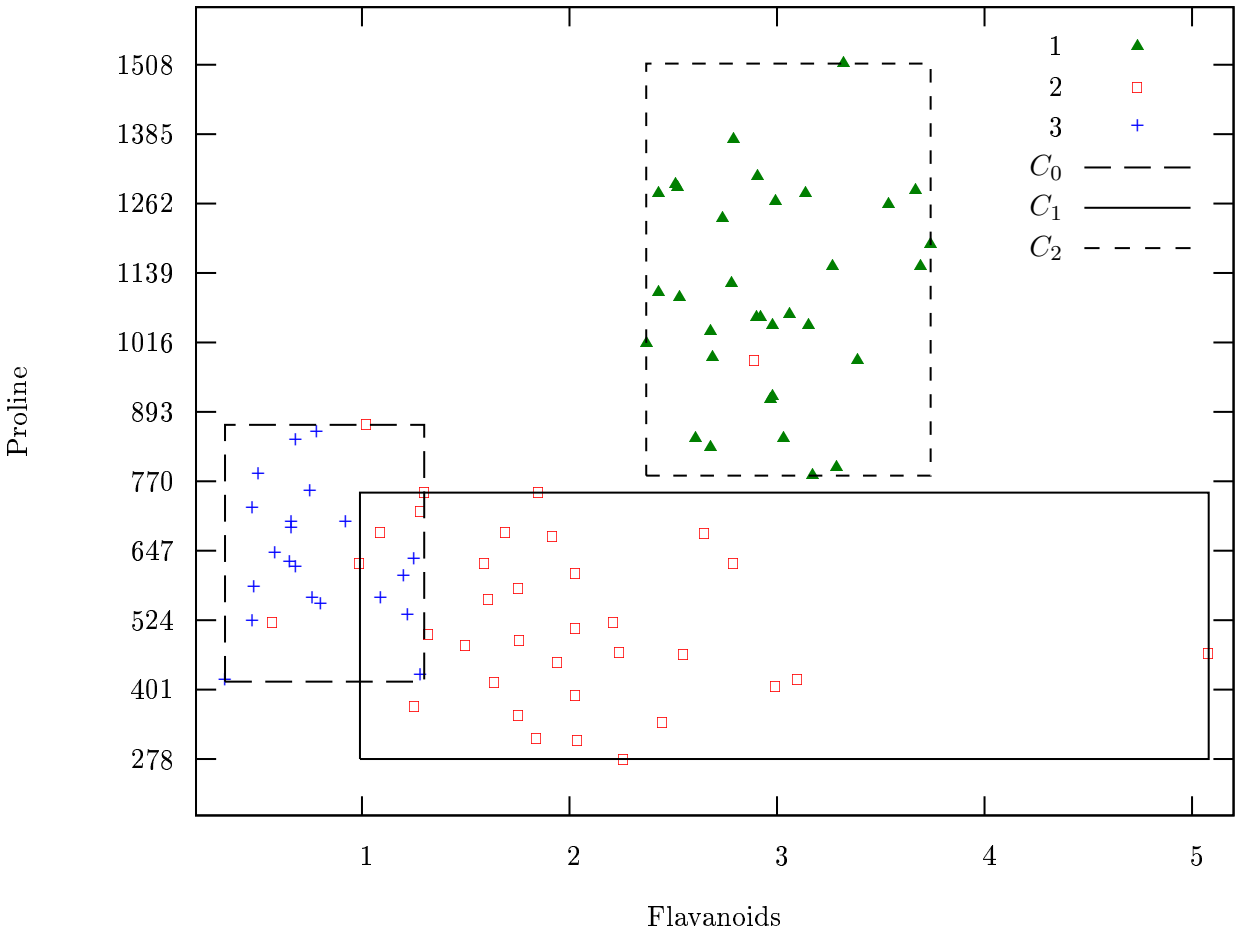


FIG. 8.4 – Meilleure projection 2D de **SuSE** sur wine.

l'objectif de C4.5 de fournir en sortie un résultat interprétable.

Ensuite, lorsque l'on calcule l'erreur de classification des deux arbres sur les données de test, l'arbre initial a un taux d'erreur de 5/89 alors que l'arbre construit avec l'aide de **SuSE** a un taux d'erreur de seulement 3/89. Les erreurs de l'arbre initial correspondent à une erreur où un élément de la classe 1 a été associé à la classe 2, et quatre erreurs où des éléments de la classe 2 ont été associés à la classe 3. D'autre part, les erreurs associées au second arbre correspondent à une erreur où un élément de la classe 1 a été associé à la classe 2, et deux erreurs où des éléments de la classe 2 ont été associés à la classe 3. La méthode a donc aidé à différencier les classes 2 et 3. Ceci semble donc confirmer notre intuition que les nouveaux attributs ajoutent au problème des informations de plus haut niveau.

8.4 Bilan

L'objectif de la nouvelle méthode d'évaluation d'algorithmes de clustering que nous proposons consiste à évaluer la capacité de ces algorithmes à fournir de l'information nouvelle et utile. Contrairement aux méthodes existantes d'évaluation, qui sont basées sur certaines notions prédéfinies de la qualité d'un clustering, notre méthode est moins subjective car elle ne présuppose qu'un objectif général d'extraction de connaissances au clustering.

Nous considérons pour cela le clustering comme un prétraitement à une autre tâche que nous sommes capables d'évaluer. Nous avons considéré ici la tâche d'apprentissage supervisé, mais nous pensons que d'autres tâches peuvent également être envisagées, comme celle par exemple d'indexer automatiquement des bases de données OLAP afin d'aider au traitement efficace de requêtes sur de telles bases.

Afin de vérifier l'intérêt général de notre méthode d'évaluation, nous nous devons de vérifier que les résultats que nous obtenons ne sont pas dépendants de l'algorithme supervisé et de la méthode de combinaison d'algorithmes supervisés et non supervisés utilisés. C'est pourquoi dans le chapitre suivant présentant les résultats que nous avons obtenus avec cette nouvelle méthode d'évaluation, nous avons considéré les deux méthodes de combinaison d'algorithmes supervisés et non supervisés présentées, ainsi que différents algorithmes supervisés :

1. C4.5 [Quinlan, 1993], qui est basé sur l'utilisation d'arbres de décision ;
2. C5 boosté [Quinlan, 2004], qui combine plusieurs classifieurs basés sur les arbres de décision ;
3. DLG [Webb and Agar, 1992], qui est basé sur l'utilisation de *moindres généralisés* ;
4. et SVM multi-classes [Tsochantaridis et al., 2005], qui utilise des combinaisons d'attributs pour définir des zones de séparation entre les données.

Chapitre 9

Expérimentations en cascade

Nous présentons dans ce chapitre les résultats des comparaisons de plusieurs algorithmes de clustering par la méthode d'évaluation en cascade que nous avons décrite dans le chapitre précédent :

1. Aléa, un algorithme qui génère des partitions aléatoires, prenant en entrée le nombre de clusters recherchés, et servant de référence ;
2. l'algorithme K-means, présenté dans la section 3.2 ;
3. LAC [Domeniconi et al., 2004], détaillé en annexes B.3.5, qui étend K-means au subspace clustering en associant à chaque centroïde de cluster un poids sur chaque dimension inversement proportionnel à la dispersion des membres du cluster sur la dimension ;
4. **SSC**, la méthode de clustering statistique présentée dans le chapitre 5 basée sur l'hypothèse d'indépendance des dimensions ;
5. et **SuSE**, également présentée dans le chapitre 5, et qui étend **SSC** en sélectionnant pour chaque cluster le sous-ensemble des attributs qui lui correspond le mieux.

Les algorithmes comparés ici utilisent donc des modèles ayant des niveaux de complexité différents. En effet, K-means utilise uniquement un centroïde pour représenter un cluster. LAC ajoute à chaque centroïde un vecteur de poids sur chaque dimension de l'espace de description. **SSC** définit une probabilité d'appartenance de chaque objet à chaque cluster et utilise un modèle gaussien (K-means et LAC n'étant pas capables de gérer les attributs catégoriels, nous ne considérerons dans ce chapitre que des données décrites par des attributs numériques). Et **SuSE** considère également un sous-ensemble des dimensions pertinentes associées à chaque cluster.

Tous ces algorithmes nécessitent de spécifier le nombre K de clusters recherchés. Comme nous en avons discuté lors de la présentation de la méthode d'évaluation en cascade dans le chapitre précédent, nous exécuterons donc plusieurs fois ces algorithmes en faisant varier K entre 2 et 10.

Parmi les cinq ensembles d'attributs qui peuvent être ajoutés aux jeux de données initiaux par la première méthode de combinaison proposée, seuls les trois premiers

sont utilisés dans ces expérimentations, parce que le quatrième ensemble dégrade les résultats et que le cinquième ne semble pas ajouter d'information supplémentaire. Or en n'utilisant pas ces deux ensembles d'attributs, nous réduisons la taille des jeux de données enrichis, ce qui permet de fournir aux apprenants supervisés des données moins complexes.

Enfin, les jeux de données étiquetés utilisés sont ceux issus de l'UCI Machine Learning Repository [Blake and Merz, 1998] qui ne contiennent que des données numériques.

9.1 Résultats initiaux

Le tableau 9.1 présente les taux d'erreurs pondérés de C4.5 sur les jeux de données initiaux de l'UCI, puis sur les jeux de données enrichis par les algorithmes de clustering correspondants. Chaque mesure correspond à une moyenne sur cinq validations croisées avec découpage du jeu de données en deux. À chaque exécution, toutes les méthodes sont confrontées au même ensemble d'apprentissage et évaluées sur le même ensemble de test. Les valeurs en gras correspondent au taux d'erreur minimum obtenu sur chaque jeu de données.

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + **SSC**	C4.5 + **SuSE**
ecoli	48.5	48.3	42.8	**40.3**	42	43.1
glass	**32.6**	40.8	35.7	37	40.4	34.9
image	4.8	6	4.8	**4.6**	**4.6**	**4.6**
iono	14.1	15.8	14.2	13.1	**9.8**	11.2
iris	7.3	7.9	6.7	**3.7**	5.1	4.7
pima	31	35	32.1	32.1	30.8	**30**
sonar	31	35.2	30	28.8	28.8	**27.2**
vowel	29.5	38.5	25	26.4	24.1	**22.2**
wdbc	5.9	6.8	4.6	3.9	5.1	**3.1**
wine	8.7	8.8	10.4	9.6	**2.7**	3.6

TAB. 9.1 – Taux d'erreur pondéré (en %) de C4.5 seul et C4.5 enrichi par les algorithmes de clustering.

Nous pouvons ainsi observer que la plupart du temps, les résultats de C4.5 sont améliorés lorsque des informations sont ajoutées à partir des résultats d'algorithmes de clustering *réels* alors qu'ils se dégradent lorsque le clustering aléatoire est utilisé. De plus, les résultats obtenus à l'aide de **SSC** et de **SuSE** sont souvent meilleurs que les résultats obtenus avec K-means et LAC.

Nous utilisons ensuite le *5×2cv F-test* [Alpaydin, 1999] pour évaluer si les résultats de deux algorithmes supervisés (dans notre cas le même algorithme, mais exécuté une fois sur le jeu de données initial, puis une seconde fois sur le jeu de données enrichi par le clustering) sont significativement différents. Le tableau 9.2 reporte ces tests entre

C4.5 seul et C4.5 enrichi par les algorithmes de clustering correspondants. Elle utilise la *p-value* associée au *5×2cv-F test*. Mais la mesure reportée est 1 moins la *p-value*. Ainsi, une valeur de 0.01 pour **SuSE** sur le jeu de données wine signifie que la probabilité que C4.5 aidé par **SuSE** surpasse C4.5 seul *par hasard* est de seulement 1%. Les valeurs en gras correspondent aux améliorations considérées comme très significatives, c'est-à-dire inférieures à 0.05.

	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + **SSC**	C4.5 + **SuSE**
ecoli	0.76	0.43	0.20	0.40	0.20
glass	0.12	0.33	0.57	0.24	0.33
image	0.24	0.61	0.61	0.54	0.67
iono	0.67	0.32	0.62	**0.02**	0.09
iris	0.53	0.81	0.65	0.43	0.22
pima	0.19	0.43	0.50	0.53	0.27
sonar	0.37	0.57	0.39	0.33	0.09
vowel	0.01	0.33	0.44	0.23	**0.03**
wdbc	0.46	0.25	**0.04**	0.63	**0.02**
wine	0.53	0.55	0.60	**0.01**	**0.01**

TAB. 9.2 – 1 - pvalue associée au 5×2cv F-test entre C4.5 seul et C4.5 enrichi par les algorithmes de clustering.

Enfin, le tableau 9.3 présente un résumé des comparaisons entre C4.5 seul et C4.5 enrichi par les algorithmes de clustering correspondants. Rappelons les mesures utilisées :

- *nb vict* est le nombre de victoires de chaque méthode ;
- *vict sign* est le nombre de victoires significatives : le nombre de fois où l'inégalité $(1 - pvalue \leq 0.05)$ est vérifiée ;
- *wilcoxon* est le *wilcoxon signed rank test* : s'il est supérieur à 1.96, alors cela signifie que l'algorithme de clustering a significativement aidé C4.5 à améliorer ses résultats ;
- et *moyenne* correspond à l'erreur pondérée moyenne (en %).

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + **SSC**	C4.5 + **SuSE**
nb vict	-	1/9	5/4	7/3	**9/1**	**9/1**
vict sign	-	0/1	0/0	1/0	2/0	**3/0**
wilcoxon	-	-2.67	-0.05	1.31	1.83	**2.56**
moyenne	21.3	24.3	20.6	20	19.3	**18.5**

TAB. 9.3 – Comparaison de C4.5 seul avec C4.5 enrichi par les algorithmes de clustering.

SuSE est donc le seul algorithme de clustering qui améliore significativement les résultats de C4.5 sur l'ensemble des jeux de données, selon le test de wilcoxon. C4.5 sur le jeu de données enrichi par **SuSE** est significativement meilleur que C4.5 sur le jeu de données initial sur trois jeux de données selon le *5×2cv-F test*. Mais comme **SuSE**, **SSC** améliore les résultats de C4.5 neuf fois sur dix, contrairement à K-means et LAC. Tous les algorithmes améliorent les résultats de C4.5 en moyenne, sauf le clustering aléatoire. Et lorsque C4.5 est combiné avec des algorithmes de clustering basés sur des modèles plus complexes, alors les taux d'erreur sont plus faibles et les améliorations sont plus significatives que lorsqu'il est combiné avec des algorithmes de clustering utilisant des modèles moins complexes.

9.2 Changements d'algorithmes supervisés

Des expérimentations avec cette méthode de combinaison d'algorithmes supervisés et non supervisés ont également été menées en utilisant deux autres algorithmes supervisés, afin d'observer si les résultats ne sont pas dépendants de l'algorithme supervisé utilisé :

1. C5 boosté 10 fois [Quinlan, 2004], pour observer si les informations ajoutées par le clustering aident également les méthodes qui combinent déjà plusieurs classifieurs ;
2. et DLG [Webb and Agar, 1992], une méthode basée sur l'utilisation de *moindres généralisés*, qui permet donc de considérer plusieurs attributs simultanément pour séparer les données, contrairement aux arbres de décision qui n'utilisent qu'un attribut à la fois.

Les expérimentations menées avec l'algorithme supervisé SVM multi-classes [Tsochantaridis et al., 2005], qui utilise des combinaisons d'attributs pour définir des zones de séparation entre les données, sont reportées à la section suivante car la gestion des attributs catégoriels par cette approche n'est pas aussi naturelle que pour C5 et DLG.

Les tableaux 9.4 et 9.5 présentent les taux d'erreur pondérés calculés dans ces deux cas. Il est alors très intéressant de remarquer que, malgré l'utilisation de différents algorithmes supervisés, les méthodes qui minimisent le taux d'erreur en validaton croisée sur les différents jeux de données restent les mêmes. En particulier, **SSC** et **SuSE** surpassent encore K-means et LAC dans la grande majorité des cas.

Les tableaux 9.6 et 9.7 résument les comparaisons d'algorithmes de clustering selon l'algorithme supervisé utilisé. Il est de nouveau intéressant de noter que l'ordre dans lequel les méthodes de clustering sont rangées reste le même quel que soit l'algorithme supervisé utilisé. Par contre, d'autres méthodes que **SuSE** deviennent significativement meilleures que l'algorithme supervisé correspondant seul. C'est par exemple le cas de **SSC** lorsque C5 est utilisé.

9.3 Changements de méthode de combinaison

Les tests ont également été menés en utilisant la seconde méthode de combinaison d'algorithmes supervisés et non supervisés présentée, c'est-à-dire en utilisant des

	C5 seul	C5 + Aléa	C5 + K-means	C5 + LAC	C5 + **SSC**	C5 + **SuSE**
ecoli	44.9	45.2	42.6	**39.9**	43.5	43.1
glass	**33.9**	43.8	35.7	37.8	35.6	34.4
image	3.3	3.9	3.2	**3**	**3**	**3**
iono	8.3	9.2	8.5	9.2	**6.6**	7.8
iris	5.6	6.3	5.3	**4.1**	4.8	4.4
pima	31.1	32.1	31.2	30.5	30.8	**29.7**
sonar	25.8	28.2	25.7	24.3	21.6	**21.3**
vowel	16	21	14.9	15	15.3	**14.5**
wdbc	4.6	4.8	3.8	4.2	4.1	**3.6**
wine	7	7.7	6.8	7.2	**2.9**	3.7

TAB. 9.4 – Taux d'erreur pondéré (en %) de C5 boosté seul et C5 boosté enrichi par les algorithmes de clustering.

	DLG seul	DLG + Aléa	DLG + K-means	DLG + LAC	DLG + **SSC**	DLG + **SuSE**
ecoli	60.8	69.7	53.3	**51.5**	54.3	54.8
glass	47.1	60.1	49.6	47.5	47.2	**46.5**
image	9.8	14.2	9.7	9.1	9.1	**8.9**
iono	22.5	32	18.9	21.2	**12.9**	15.4
iris	9.1	15.6	8	**5.1**	6.4	6.5
pima	40.6	43.7	39.1	39.9	38.6	**38.4**
sonar	45.5	45.9	43.5	42.9	40.8	**38**
vowel	50.5	64.5	44.5	45	44	**42.7**
wdbc	9.2	10.7	7.7	7.8	7.1	**6.9**
wine	18.5	22.1	18.1	16.3	**6.9**	8

TAB. 9.5 – Taux d'erreur pondéré (en %) de DLG seul et DLG enrichi par les algorithmes de clustering.

	C5 seul	C5 + Aléa	C5 + K-means	C5 + LAC	C5 + **SSC**	C5 + **SuSE**
nb vict	-	0/10	7/3	7/3	**9/1**	**9/1**
vict sign	-	0/0	0/0	0/0	0/0	0/0
wilcoxon	-	-2.88	1.41	1.31	**2.04**	**2.67**
moyenne	18	20.2	17.8	17.5	16.8	**16.5**

TAB. 9.6 – Comparaison de C5 boosté seul avec C5 boosté enrichi par les algorithmes de clustering.

	DLG seul	DLG + Aléa	DLG + K-means	DLG + LAC	DLG + **SSC**	DLG + **SuSE**
nb vict	-	0/10	9/1	9/1	9/1	**10/0**
vict sign	-	0/2	1/0	**2/0**	**2/0**	**2/0**
wilcoxon	-	-2.88	**2.15**	**2.77**	**2.77**	**2.88**
moyenne	31.4	37.9	29.2	28.6	26.7	**26.6**

TAB. 9.7 – Comparaison de DLG seul avec DLG enrichi par les algorithmes de clustering.

classifieurs locaux sur chaque groupe de données identifié par les algorithmes de clustering. Les tableaux 9.8 et 9.9 reportent les taux d'erreur obtenus lorsque C4.5 ou SVM sont utilisés en tant qu'apprenants supervisés, et les tableaux 9.10 et 9.11 résument les résultats.

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + **SSC**	C4.5 + **SuSE**
ecoli	48.5	51.1	39.3	**32.1**	39.3	43.4
glass	**32.6**	46.4	33.9	32.8	33.1	33.1
image	4.8	6.8	4.8	4.8	4.3	**4**
iono	14.1	15.9	13.6	11.2	**10.1**	11.1
iris	7.3	7.6	5.5	**3.1**	4.9	4
pima	31	31.4	30.3	32.2	30.2	**28.9**
sonar	31	32.9	26.5	24.5	25.4	**22.3**
vowel	29.5	41.8	26.1	26.3	26	**25**
wdbc	5.9	6.5	3.5	4	3.5	**2.7**
wine	8.7	7.5	10.4	10	**2.2**	3

TAB. 9.8 – Taux d'erreur pondéré (en %) de C4.5 seul et C4.5 enrichi par les algorithmes de clustering selon la seconde méthode de combinaison.

Nous observons donc encore dans ces cas que, malgré l'utilisation d'une autre méthode de combinaison d'algorithmes supervisés et non supervisés, l'ordre dans lequel les méthodes de clustering sont rangées reste le même. En particulier, **SSC** et **SuSE** fournissent encore de meilleurs résultats que K-means et LAC.

Lorsque SVM est utilisé comme apprenant supervisé, K-means semble surpasser LAC, mais ce résultat dépend fortement des mauvais résultats obtenus par LAC sur le jeu de données *ionosphere*. Notons également que nous avons utilisé les paramètres par défaut de SVM, parce que notre objectif ici n'est pas l'amélioration des résultats d'algorithmes supervisés à l'aide d'algorithmes non supervisés, mais bien l'évaluation des algorithmes non supervisés à l'aide d'algorithmes supervisés. Donc même si les paramètres utilisés pour SVM ne sont pas optimisés, tous les algorithmes de clustering sont mis au même niveau car ils utilisent tous le même algorithme supervisé.

	SVM seul	SVM + Aléa	SVM + K-means	SVM + LAC	SVM + **SSC**	SVM + **SuSE**
ecoli	52.9	63.6	36.5	**35.9**	40.1	36.8
glass	**3.0**	10.3	6.3	5.4	4.3	4.4
image	4.9	7.8	4.7	4.5	4.4	**4**
iono	20.2	23.3	15.5	18.9	**13.9**	14.6
iris	4.4	4.9	2.9	**2.0**	2.7	2.5
pima	41.5	37.3	33.1	35	32	**31.2**
sonar	28.4	30.3	22.2	20.2	18.1	**17.8**
vowel	55.2	59.1	**24.3**	24.7	25.5	26.2
wdbc	13.8	15.2	4	5.2	3.8	**3.7**
wine	6.8	9.3	6.8	6.8	**2.5**	3.3

TAB. 9.9 – Taux d'erreur pondéré (en %) de SVM seul et SVM enrichi par les algorithmes de clustering selon la seconde méthode de combinaison.

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + **SSC**	C4.5 + **SuSE**
nb vict	-	1/9	7/2	6/3	**9/1**	**9/1**
vict sign	-	0/3	0/0	0/0	1/0	**3/0**
wilcoxon	-	-2.46	1.1	1.2	**2.72**	**2.77**
moyenne	21.3	24.8	19.4	18.1	17.9	**17.8**

TAB. 9.10 – Comparaison de C4.5 seul avec C4.5 enrichi par les algorithmes de clustering selon la seconde méthode de combinaison.

	SVM seul	SVM + Aléa	SVM + K-means	SVM + LAC	SVM + **SSC**	SVM + **SuSE**
nb vict	-	1/9	8/1	8/1	**9/1**	**9/1**
vict sign	-	1/4	3/0	3/0	4/0	**7/0**
wilcoxon	-	-1.99	1.48	1.43	**2.6**	**2.6**
moyenne	23.1	26.1	15.6	15.9	14.7	**14.4**

TAB. 9.11 – Comparaison de SVM seul avec SVM enrichi par les algorithmes de clustering selon la seconde méthode de combinaison.

	Aléa	K-means	LAC	**SSC**	**SuSE**
ecoli	0.352	**0.792**	0.776	0.781	0.776
glass	0.364	0.565	0.557	0.532	**0.568**
image	0.280	0.597	0.595	0.653	**0.733**
iono	0.524	0.734	0.615	**0.759**	0.686
iris	0.432	0.861	0.875	**0.952**	0.940
pima	0.526	**0.662**	0.641	0.523	0.622
sonar	0.535	**0.584**	0.564	0.545	0.556
vowel	0.186	0.402	**0.425**	0.378	0.380
wdbc	0.531	0.959	**0.964**	0.935	0.932
wine	0.468	0.730	0.741	**0.978**	0.944

TAB. 9.12 – F-mesure calculant la correspondance entre les labels de clusters et les labels de classes.

9.4 Bilan

Sur l'ensemble des expériences que nous avons menées avec la méthode d'évaluation en cascade d'algorithmes de clustering que nous avons proposée, nous avons pu observer que quels que soit l'algorithme supervisé et la méthode de combinaison d'algorithmes supervisés et non supervisés utilisés, l'ordre dans lequel les méthodes de clustering sont rangées reste le même. Ce résultat montre donc la stabilité de cette méthode d'évaluation.

En particulier, les expérimentations ont mis en avant le fait que les méthodes de clustering basées sur l'utilisation de modèles plus complexes surpassent les méthodes utilisant des modèles moins complexes. Ce résultat n'est pas surprenant mais montre le comportement cohérent de notre méthode d'évaluation. Il met également en avant l'intérêt de la méthode de sélection d'attributs que nous avons proposée dans l'algorithme **SuSE** puisque **SuSE** obtient de meilleurs résultats que **SSC**.

À titre de comparaison, les tableaux 9.12 et 9.13 reportent la *F-mesure* et l'*Entropie* (mesures de la qualité externe de partitions, présentées dans la section 2.6) calculées entre les clusters obtenus par les différents algorithmes de clustering et les classes initiales des différents problèmes, permettant d'évaluer leur correspondance. Les valeurs en gras correspondent aux valeurs maximales obtenues sur chaque jeu de données pour le tableau 9.12, et aux valeurs minimales pour le tableau 9.13.

La première observation que l'on peut faire à partir de ces deux tableaux est que les deux mesures ne sont pas d'accord sur quelles méthodes de clustering ont la meilleure correspondance entre les labels de clusters et les labels de classes sur chaque jeu de données. De plus, nous pouvons remarquer qu'il n'y a pas de relation directe entre les méthodes de clustering qui optimisent ces valeurs et celles qui aident le mieux les méthodes supervisées à améliorer leurs performances.

Ainsi, nous mettons en avant le fait que notre méthode d'évaluation ne se contente pas d'évaluer les correspondances obtenues par les différentes méthodes de clustering

	Aléa	K-means	LAC	**SSC**	**SuSE**
ecoli	1.987	**0.537**	0.644	0.676	0.689
glass	1.836	1.214	1.181	1.351	**1.076**
image	2.542	1.368	1.431	0.952	**0.922**
iono	0.900	0.509	0.671	**0.251**	0.377
iris	1.345	0.134	**0.091**	0.149	0.153
pima	0.925	0.776	0.784	**0.739**	0.746
sonar	0.893	0.820	0.734	**0.713**	0.758
vowel	3.319	**2.016**	2.039	2.100	2.127
wdbc	0.929	**0.125**	0.220	0.136	0.153
wine	1.364	0.688	0.727	0.133	**0.044**

TAB. 9.13 – Entropie calculant la correspondance entre les labels de clusters et les labels de classes.

entre les labels de clusters et les labels de classes, mais qu'elle capture bien la capacité des méthodes de clustering à fournir des informations de plus haut niveau.

Enfin, de tels tableaux de mesures ne fournissent aucune information objective sur l'intérêt des méthodes de clustering, alors qu'avec la méthode d'évaluation que nous proposons, nous pouvons observer dans quelle mesure les informations apportées par le clustering sont utiles, et en particulier si ces nouvelles informations sont significatives.

Quatrième partie

Conclusions et perspectives

Articulées autour de la problématique de l'apprentissage non supervisé, ou clustering, nos recherches se sont finalement situées à plusieurs niveaux.

Face à la quantité importante d'applications et de méthodes différentes existant dans ce cadre, nous avons tenté, dans la partie état de l'art de cette thèse, de présenter un tour d'horizon des méthodes les plus classiques en détaillant leurs atouts et faiblesses ainsi que les cas d'application qui leur correspondent le mieux. Les tableaux de caractéristiques des méthodes et le bilan de la partie permettent ainsi au lecteur qui recherche une méthode de clustering pour un cas d'application particulier d'identifier la famille de méthode la plus appropriée dans son cas.

Restreignant ensuite la recherche au cas où l'apprentissage non supervisé est utilisé dans un cadre d'extraction de connaissances, nous avons dans un premier temps étudié comment présenter les résultats de manière compréhensible à l'utilisateur, et nous nous sommes alors naturellement tournés vers les règles qui se sont avérées tout à fait appropriées dans ce cadre. Les résultats produits sont ainsi facilement interprétables par l'utilisateur. Et les règles sont également des structures tout à fait appropriées pour caractériser les groupes présents dans les données dans de nombreux cas, et en particulier lorsque des chevauchements existent entre groupes, comme dans les cas présentés dans la figure 9.1 par exemple, alors que des représentations classiques comme les centres de groupes ou des arbres de décision ne seront pas appropriés pour représenter les groupes dans de tels cas.

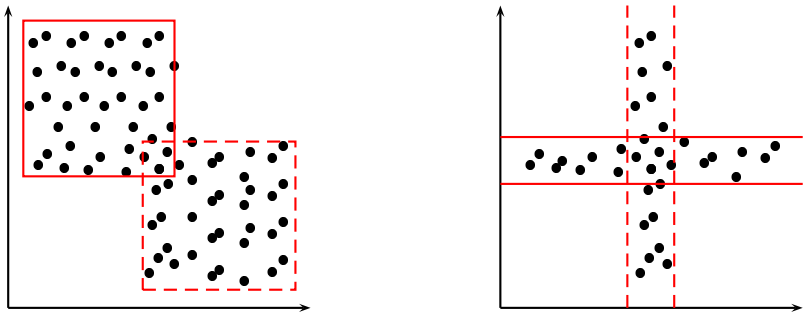


FIG. 9.1 – Intérêt de l'utilisation de règles pour représenter les clusters.

Un autre avantage des règles est que leur manipulation permet des traitements efficaces. Supposer l'indépendance entre dimensions permet d'accélérer la méthode, et tester l'appartenance d'un objet à une règle, ou l'apparition de nouveaux objets dans une règle lorsque l'une de ses clauses sur un attribut est supprimée, est plus rapide qu'avec d'autres structures comme les ellipsoïdes par exemple.

Confrontés aux données réelles pouvant contenir de nombreux attributs non pertinents, nous nous sommes ensuite situés dans le cadre du subspace clustering, dont l'objectif est d'identifier non seulement les groupes présents dans les données, mais également les sous-ensembles d'attributs spécifiques sur lesquels ces groupes sont définis. C'est le problème de la *contextualisation*, dont la prise en compte s'inscrit par ailleurs tout à fait dans l'objectif de description compréhensible des résultats puisqu'un

ensemble réduit d'attributs est alors utilisé pour caractériser les groupes.

Recherchant d'autre part à mettre en œuvre une méthode ne nécessitant aucune connaissance a priori de la part de l'utilisateur, nous avons d'abord envisagé une méthode divisive nommée **Tuareg** qui s'est révélée intéressante sur les différentes problématiques que nous venons d'évoquer, mais limitée dans les cas où un seul attribut ne suffit pas pour différencier différents groupes. Nous nous sommes alors tournés vers les méthodes basées sur l'utilisation de modèles probabilistes, qui nous ont permis d'atteindre notre objectif en nous permettant de transformer le problème difficile de la spécification des paramètres inhérent à toute méthode de subspace clustering en problème de sélection de modèle dans le cadre probabiliste. La méthode proposée, nommée **SuSE**, s'est ainsi révélée elle aussi tout à fait intéressante face aux différentes problématiques évoquées, mais capable, elle, d'identifier des groupes même si leurs différences ne sont visibles que dans des espaces de dimensionalité supérieure à un.

Une méthode de visualisation graphique des règles produites nous a finalement permis d'observer de façon tout à fait naturelle les spécificités des groupes identifiés ainsi que leurs différences principales. Dans le cas de la figure 9.2 par exemple, on observe ainsi aisément la présence de quatre groupes dans les données : l'un défini par des valeurs élevées sur X, un autre par des valeurs élevées sur Y, un troisème par des valeurs moyennes sur X et sur Y, et enfin un dernier groupe pour lesquels les attributs X et Y ne sont pas discriminants.

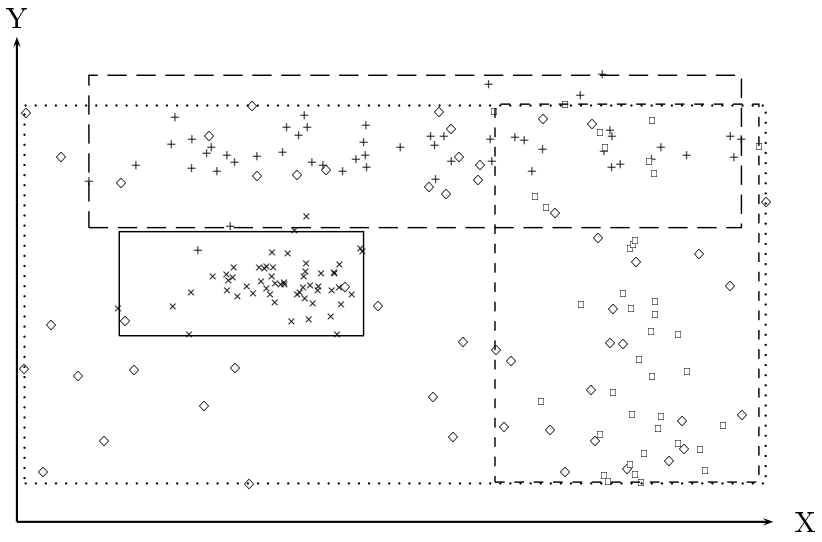


FIG. 9.2 – Visualisation graphique des résultats.

La méthode globale que nous proposons permet donc finalement de réduire, par étapes successives, le nombre d'attributs considérés pour caractériser les groupes ciblés. Tout d'abord, seuls les attributs pertinents des groupes sont pris en compte lors de leur détection par la méthode de subspace clustering proposée, **SuSE**. Puis une seconde étape de description minimale des règles associées aux groupes est ensuite utilisée afin

de réduire davantage le nombre d'attributs utilisés dans la présentation des résultats. Une dernière étape de projection de ces règles dans un espace à deux dimensions va finalement permettre d'observer visuellement les spécificités et différences des différents groupes détectés. Cette dernière étape permet par ailleurs de pallier aux possibles limites de l'utilisation des règles pour représenter les groupes, en particulier lorsque l'hypothèse d'indépendance des attributs n'est pas respectée, puisque de telles particularités peuvent être facilement identifiées visuellement.

Confrontés ensuite, lors du challenge INEX 2006, au cas particulier où les données sont des documents XML basés sur une structure arborescente, nous avons pu observer l'intérêt de transformer ces arbres en attributs-valeurs afin de bénéficier des atouts des méthodes existantes capables de traiter des données sous forme d'attributs-valeurs. C5 boosté a ainsi obtenu les meilleurs résultats du challenge en apprentissage supervisé en utilisant de telles transformations. **SuSE**, pour sa part, a obtenu de très bons résultats dans le cas de l'apprentissage non supervisé, et s'est par ailleurs démarqué de ses concurrents par sa capacité à fournir également en sortie des résultats facilement interprétables par l'utilisateur.

Confrontés enfin au problème de l'évaluation de l'intérêt de la méthode proposée, nous nous sommes rendus compte que même des bonnes performances sur données artificielles et des résultats considérés comme intéressants par des experts dans le cas d'applications réelles n'étaient pas suffisants pour généraliser son intérêt dans n'importe quel cas d'application. Pour pallier à cette limite, nous avons donc proposé de replacer le clustering dans son contexte d'utilisation afin d'évaluer son intérêt dans ce cadre particulier. Et puisque celui-ci est souvent utilisé en prétraitement d'un apprentissage supervisé, c'est dans ce cadre que nous avons mené nos expériences, et ainsi montré l'intérêt de notre approche.

Voyons maintenant plus en détails les conclusions que nous pouvons tirer des recherches que nous avons menées sur ces différents points, et des méthodes que nous avons proposées, ainsi que les perspectives ouvertes par ces travaux.

Chapitre 10

Méthodes de clustering

10.1 Le clustering

Le clustering est un domaine de recherche qui est étudié depuis de nombreuses années dans différentes communautés : machine learning, data mining, pattern recognition, statistiques, etc.

Dans la partie état de l'art de cette thèse, nous ne prétendons pas avoir fourni une liste exhaustive de l'ensemble des méthodes de clustering qui ont été proposées, mais plutôt avoir fourni une vue d'ensemble des principales problématiques existantes et techniques utilisées dans ce cadre. Plusieurs compléments d'informations peuvent être trouvés dans [Jain et al., 1999, Jain et al., 2000, Berkhin, 2002].

Comme nous l'avons vu, il existe différents types d'applications au clustering. Il peut être utilisé pour compresser un ensemble de données, pour indexer de façon automatique des bases de données OLAP, pour discrétiser un ensemble de données, pour identifier des communautés de clients aux comportements similaires, pour organiser un ensemble de documents, ou pour appréhender un jeu de données en faisant émerger certains sous-concepts présents dans les données.

Face à ces différents champs d'applications, et face à l'ensemble très hétérogène des bases de données qui peuvent être analysées, il n'est pas raisonnable de penser trouver une méthode unique qui surpasserait toutes les autres. En effet, lorsque le clustering a pour objectif la compression des données, alors une méthode capable de fournir un résultat compréhensible n'est pas nécessaire et considérer des transformations des données ne pose pas de problème. Au contraire, lorsque le clustering a pour objectif d'appréhender un jeu de données en faisant émerger certains sous-concepts dans les données, alors il devient nécessaire d'utiliser une méthode capable de présenter de manière compréhensible les résultats, et une transformation des données n'est pas souhaitable. De même, si le clustering est utilisé pour effectuer de la segmentation d'images, alors une méthode capable de détecter des clusters de forme variée est nécessaire, alors que si le clustering est utilisé afin d'indexer des bases de données OLAP, utiliser une méthode capable de fournir en sortie un ensemble de clusters décrits par des règles est plus approprié.

Ainsi, de manière plus générale, lorsque l'on souhaite utiliser une méthode de clus-

tering pour une application particulière, il faut tout d'abord spécifier à quels types de bases de données on est confronté, et quel type de résultat on en attend. Ensuite, et c'est ce que nous avons essayé de mettre en place dans cette thèse en associant à chaque méthode présentée un tableau de ses caractéristiques principales, il faut choisir parmi l'ensemble des méthodes de clustering existantes celle qui correspond le mieux à son problème.

C'est pourquoi nous nous sommes concentré dans nos travaux sur un champ d'application particulier : celui de l'extraction de connaissances, c'est-à-dire de l'identification de sous-concepts pouvant émerger par l'analyse non supervisée de données hétérogènes. Dans ce cadre, nous avons identifié plusieurs problématiques importantes :

1. minimiser le nombre de connaissances a priori requises de la part de l'utilisateur ;
2. être capable de faire face à des bases de données hétérogènes pouvant contenir des attributs non pertinents, du bruit et des données manquantes ;
3. fournir un résultat compréhensible par l'utilisateur en un temps raisonnable.

Nous nous sommes donc naturellement intéressé au subspace clustering, qui prend en compte la possible présence d'attributs non pertinents, et par là même permet d'obtenir des descriptions réduites des clusters, et permet également potentiellement d'atteindre des complexités réduites en ne considérant pas l'ensemble des dimensions de description des objets, qui peut s'avérer très important dans certains cas.

Une dernière problématique importante en clustering est celle de l'évaluation des résultats. Là encore, comme la notion même de l'intérêt d'un résultat de clustering peut varier selon l'application visée, il n'est pas raisonnable de penser qu'une méthodologie surpassera toutes les autres. Au contraire, les différentes techniques existantes sont complémentaires et doivent être utilisées en tant que tel. Ainsi, démarrer en confrontant une méthode de clustering à des données générées artificiellement permet d'évaluer sa robustesse dans certains cas précis qui sont contrôlés. Confronter la méthode à des données réelles est ensuite nécessaire afin d'observer son efficacité pratique. Enfin, nous pensons que la méthode d'évaluation que nous avons proposée permet d'évaluer la pertinence d'une méthode dans un certain cadre (celui du prétraitement pour un algorithme supervisé dans le cas présenté) et face à un ensemble de jeux de données indépendants.

Enfin, une autre problématique émergente en clustering concerne l'application des méthodes existantes au cas des données représentées autrement que par des ensembles d'attributs-valeurs. Lors de nos travaux, nous avons proposé une méthode originale permettant de considérer des données présentées sous forme arborescente, mais il s'avère aujourd'hui de plus en plus intéressant d'être capable de prendre également en compte des données présentées sous la forme de courbes ou bien des données symboliques [Diday and Vrac, 2005].

10.2 Algorithme Tuareg

La première méthode que nous avons proposée dans ce cadre, que nous avons nommée **Tuareg**, s'est révélée très intéressante sur de nombreux points évoqués précédemment. En effet, **Tuareg** est d'une complexité tout à fait raisonnable, il ne nécessite

aucune connaissance a priori de la part de l'utilisateur, et fournit en sortie un résultat compréhensible sous forme de règles définies par un minimum de dimensions parmi les plus pertinentes.

Basé sur l'hypothèse que pour tout couple de clusters distincts, il existe au moins une dimension de projection sur laquelle ces deux clusters sont bien séparés, **Tuareg** n'est cependant pas capable de détecter des clusters qui seraient bien séparés dans l'espace complet de description des objets mais dont la séparation ne serait visible sur aucune projection à une seule dimension.

Remplacer l'algorithme de partitionnement élémentaire par une méthode plus élaborée qu'une simple division au niveau de l'écart le plus important existant dans la projection des données sur chaque dimension (en utilisant par exemple un clustering probabiliste sur ces projections) permettrait de résoudre une partie du problème. De même, considérer des projections des données sur plus d'une dimension permettrait également de limiter le problème. Cependant, de telles modifications de la méthode auraient pour conséquence d'augmenter de façon importante sa complexité.

Finalement, nous pensons donc que **Tuareg** peut se révéler très intéressant s'il est utilisé en première phase d'une analyse de données car il est tout à fait efficace pour cibler des groupes bien séparés dans les données, alors que d'autres méthodes plus élaborées nécessiteraient des temps d'exécution plus importants pour identifier les mêmes groupes de données.

10.3 Algorithme SuSE

La seconde méthode que nous avons proposée, nommée **SuSE**, regroupe également les avantages de ne nécessiter aucune connaissance a priori de la part de l'utilisateur et de fournir un résultat compréhensible sous forme de règles définies par un minimum de dimensions parmi les plus pertinentes. Sa complexité, supérieure à celle de **Tuareg**, est tout de même réduite car **SuSE** ne considère que des sous-ensembles de dimensions associées aux clusters, au lieu de considérer l'ensemble complet de description des objets. Efficace face à des bases de données bruitées pouvant contenir des données manquantes, **SuSE** a démontré son intérêt dans le cas d'applications réelles.

L'idée d'utiliser des modèles probabilistes pour effectuer un clustering des données basé sur l'utilisation de règles a déjà été étudiée dans [Pelleg and Moore, 2001]. Il existe plusieurs différences entre notre méthode et la leur. La première apparaît dans la modélisation : au lieu de supposer la distribution gaussienne sur une dimension numérique, les auteurs la supposent uniforme à l'intérieur d'un intervalle donné, et utilisent une *queue* de distribution aux bords de cet intervalle, dépendant d'un paramètre σ qui évolue au cours de l'algorithme. Cette différence se retrouve ensuite dans la méthode finale de clustering. En particulier, leur méthode n'est pas capable de mettre à jour son modèle de façon incrémentale, alors que la nôtre peut s'adapter à la présentation de nouveaux objets. De plus, nous avons proposé une méthode efficace permettant d'effectuer de la sélection d'attributs pendant l'apprentissage alors que cette problématique n'a pas été prise en compte dans leurs travaux.

L'idée d'utiliser l'hypothèse d'indépendance des dimensions pour effectuer de la sélection d'attributs a elle déjà été étudiée dans [Law et al., 2004]. Mais la méthode décrite par les auteurs diffère également de la nôtre sur certains points. Tout d'abord, la sélection d'attributs qu'ils proposent est globale à l'ensemble des clusters alors que les attributs sélectionnés par **SuSE** sont localement pertinents pour les clusters. De plus, dans [Law et al., 2004], les auteurs ajoutent plusieurs paramètres à leur modèle pour effectuer une pondération sur les attributs alors que dans **SuSE**, le nombre de paramètres du modèle est réduit grâce à une sélection rigide d'attributs. Ainsi, notre méthode de sélection d'attributs accélère l'algorithme alors que dans [Law et al., 2004], les auteurs sont confrontés à des problèmes de complexité en temps et en espace.

Par ailleurs, nous avons intégré effectivement la problématique catégorielle qui n'était évoquée qu'à titre de perspectives dans les travaux menés dans [Pelleg and Moore, 2001] et [Law et al., 2004]. Nous avons également proposé une méthode originale de sélection d'attributs permettant de présenter les résultats sous forme de règles définies par un minimum d'attributs parmi les plus pertinents, ainsi qu'une méthode de visualisation graphique de ces règles dont nous avons pu observer l'intérêt dans différents cas d'application réels. À ce niveau, nous pensons qu'une étude plus approfondie pourrait être menée concernant le critère de sélection des projections en deux dimensions les plus appropriées pour la visualisation des résultats. Une application de cette technique à des projections sur trois dimensions pourrait également être envisagée.

Nous avons également pu observer l'intérêt d'utiliser un critère d'arrêt de type K-means lors de l'optimisation des modèles par l'algorithme EM, afin d'accélérer la méthode. Pour poursuivre la recherche dans ce sens, il semble intéressant de s'inspirer des travaux présentés dans [Bradley et al., 1998] qui traitent de l'accélération de l'algorithme EM dans le cas général. De même, les travaux de [Jollois and Nadif, 2004] dans ce cadre peuvent également être intégrés au système dans le but d'accélérer la méthode.

Par ailleurs, il serait intéressant de poursuivre cette recherche afin de trouver une méthode efficace permettant d'atteindre la valeur optimum du critère BIC, identifiant le nombre de clusters et le nombre de dimensions pertinentes des clusters les plus appropriés, sans avoir à tester tous les modèles possibles pour ces deux paramètres. Une recherche par descente de gradient semble pour cela tout à fait appropriée au vu de la figure 7.5.

Enfin, nous avons pu observer l'intérêt de l'adaptation de notre méthode au cas des données semi-structurées, par leur transformation en attributs-valeurs. Nous pensons qu'il est possible d'aller plus loin dans cette technique de transformation. Par exemple, il est possible de considérer comme nouveaux attributs certaines fourches présentes dans les arbres, ou bien d'identifier dans quelle partie des arbres les labels ou relations entre labels sont présents. Mais comme nous avons pu le constater dans la partie expérimentale de ces travaux, la méthode de transformation que nous avons proposée génère déjà de nombreux attributs, et les résultats obtenus avec de tels attributs sont déjà très bons. Afin de prendre en compte de telles différences possibles entre les arbres, une attention particulière devrait alors être portée sur la recherche d'un bon compromis entre le nombre de nouveaux attributs créés et l'information qu'ils contiennent.

Chapitre 11

Évaluation en cascade

11.1 Méthode d'évaluation du clustering

Une autre contribution importante de nos travaux concerne la proposition d'une nouvelle méthode d'évaluation d'algorithmes de clustering. Plus globale, objective et quantitative que les méthodes existantes, notre approche consiste à comparer les résultats d'un algorithme supervisé lorsqu'il est (ou pas) aidé par de l'information issue des résultats d'un algorithme de clustering. Nous avons considéré différents algorithmes supervisés et différentes façons de combiner les résultats d'algorithmes supervisés et non supervisés. Les expérimentations ont alors mis en avant le fait que l'ordre dans lequel les méthodes de clustering sont rangées reste le même quels que soient l'algorithme supervisé et la méthode de combinaison utilisés. Ceci montre donc la stabilité de la nouvelle méthode d'évaluation que nous proposons.

Les expérimentations ont également mis en avant que les méthodes de clustering basées sur l'utilisation de modèles plus complexes surpassent les méthodes utilisant des modèles moins complexes. Ce résultat n'est pas surprenant mais montre le comportement cohérent de notre méthode d'évaluation. Et cela montre en particulier l'intérêt de notre méthode de sélection rigide d'attributs dans **SuSE**, puisque la méthode n'utilisant pas cette sélection rigide d'attributs, nommée **SSC**, s'avère moins intéressante dans le cadre de cette évaluation.

Dans des travaux futurs, il semble intéressant de trouver d'autres tâches aussi objectives que l'apprentissage supervisé et pour lesquelles le clustering serait un prétraitement intéressant, afin de tester notre méthode d'évaluation dans un tel cadre. Cela pourrait être le cas par exemple dans une tâche d'indexation automatique de bases de données OLAP. En effet, si le clustering aide à indexer de manière efficace de telles bases de données, alors l'exécution de requêtes sur ces bases pourrait s'en trouver accélérée. Dans ce cas, on pourrait tout à fait envisager d'utiliser le clustering dans l'objectif d'indexer de manière automatique un ensemble de bases, sur lesquelles seraient ensuite exécutées un certain nombre de requêtes, dont le temps d'exécution permettrait de mesurer l'intérêt du clustering dans ce cadre.

11.2 Combinaison de classifieurs

Nous avons également montré que les résultats des algorithmes supervisés sont améliorés lorsqu'ils utilisent de l'information issue d'algorithmes de clustering (autres qu'aléatoires). Nous postulons que les algorithmes supervisés peuvent bénéficier de cette information car elle est de nature très différente. En particulier, les algorithmes de clustering peuvent aider les algorithmes supervisés à spécialiser leurs traitements en fonction de différentes zones spécifiques de l'espace de description des objets. Ils peuvent également aider les algorithmes supervisés à décrire des zones de décision plus complexes. Il semble donc intéressant de poursuivre cette recherche dans le cadre plus général de la combinaison de classifieurs lorsque l'un des classifieurs est non supervisé.

Comme nous l'avons vu dans nos expérimentations, il semble qu'utiliser le clustering pour partitionner l'espace des objets puis exécuter des apprentissages supervisés indépendants sur chaque groupe d'objets créé produise de meilleurs résultats que d'enrichir les jeux de données en créant de nouveaux attributs puis d'exécuter un apprentissage supervisé sur le jeu de données enrichi (comparaison des résultats des tableaux 9.3 et 9.10). Mais cela peut être la conséquence de la méthode que nous avons utilisée pour créer de nouveaux attributs, qui accroît de manière significative la taille du jeu de données. Il serait donc intéressant de creuser ce point dans de futures recherches.

Chapitre 12

Bilan

En résumé, une notion est revenue de manière centrale tout au long de cette thèse : celle de la prise en compte du **contexte** en apprentissage :

- le contexte dans lequel les groupes présents dans les données sont créés doit être pris en compte, c'est-à-dire que la méthode d'apprentissage doit être capable de détecter pour chacun des groupes ciblés l'ensemble des attributs qui sont pertinents pour sa caractérisation ;
- la prise en compte du contexte dans lequel les règles associées à ces groupes sont définies permet pour sa part de fournir en sortie un ensemble de règles décrites par un minimum d'attributs, permettant ainsi à l'utilisateur d'interpréter plus facilement le résultat produit ;
- et la prise en compte du contexte dans lequel l'apprentissage non supervisé est utilisé permet de lui offrir un cadre pour son évaluation.

Un autre point important dans nos recherches concerne l'utilisation des **règles** comme structure pour représenter les groupes ciblés dans les données. Les régles ont l'intérêt particulier d'être tout à fait naturelles pour l'homme. Elles sont souvent utilisées comme base pour toute représentation de connaissances. C'est par des règles que les hommes définissent leur environnement, leurs relations et leur organisation. Côté méthode d'apprentissage, les règles permettent par ailleurs une description riche des résultats, ainsi qu'une manipulation facilitée par rapport à nombre d'autres structures.

Enfin l'étude que nous avons menée dans le cadre de l'**évaluation en cascade** d'algorithmes de clustering nous suggère une relation forte entre méthodes d'apprentissage supervisé et non supervisé. Nous avons pu observer que leur combinaison permettait une évaluation plus globale, objective et quantitative des méthodes non supervisées, mais aussi que les méthodes supervisées pouvaient tirer avantage de telles combinaison afin d'améliorer leurs performances. Des recherches dans ce domaine semblent par conséquent tout à fait intéressantes et prometteuses.

Les règles existent
tant qu'aucun contre-exemple
ne vient mettre en peine leur existence.

Laurent Candillier

Cinquième partie

Annexes

Cette dernière partie est consacrée à la présentation plus détaillée de certaines notions et méthodes qui ont été évoquées précédemment.

La première annexe présente d'abord plusieurs mesures utilisées classiquement pour définir la similarité entre données et l'optimalité d'une solution, qui constituent la base de toute méthode de clustering. Nous présentons ensuite différentes méthodes proposées pour faire face aux larges bases de données, ainsi que plusieurs algorithmes efficaces dans ce cadre.

L'annexe suivante fournit davantage de détails sur les différentes techniques qui ont été proposées pour faire face aux attributs non pertinents qui décrivent les données. Nous présentons d'abord plusieurs approches cherchant l'espace de description le plus approprié au traitement des données, puis les principaux algorithmes de subspace clustering développés récemment.

Enfin les deux dernières annexes reviennent sur certains détails dans la mise en œuvre des algorithmes **Tuareg** et **SuSE** que nous avons proposés.

Annexe A

Algorithmes de clustering

A.1 Notions de similarité

L'une des problématique centrale du clustering est de définir la notion de similarité entre les données et de la qualité d'une solution. Il existe trois concepts de similarité en clustering :

1. la similarité entre objets : à maximiser pour deux objets appartenant au même cluster, et à minimiser pour deux objets appartenant à des clusters différents ;
2. la similarité entre un objet et un cluster : à maximiser si l'objet est associé au cluster pour une bonne *cohésion interne* du cluster ;
3. et la similarité entre clusters : à minimiser pour une bonne *isolation externe* des clusters.

A.1.1 Similarités entre objets

Typiquement, la similarité entre objets est évaluée par une fonction de distance définie entre paire d'objets. Les mesures de distance les plus courantes entre deux objets $\overrightarrow{x_i}$ et $\overrightarrow{x_j}$ sont les suivantes :

- la distance de Manhattan :

$$dist_1(\overrightarrow{x_i}, \overrightarrow{x_j}) = |\overrightarrow{x_i} - \overrightarrow{x_j}| = \sum_{d=1}^{M} |x_{id} - x_{jd}|$$

- la distance euclidienne :

$$dist_2(\overrightarrow{x_i}, \overrightarrow{x_j}) = \|\overrightarrow{x_i} - \overrightarrow{x_j}\|_2 = \sqrt{\sum_{d=1}^{M} (x_{id} - x_{jd})^2}$$

- la distance de Minkowski, qui généralise les deux précédentes :

$$dist_p(\overrightarrow{x_i}, \overrightarrow{x_j}) = \|\overrightarrow{x_i} - \overrightarrow{x_j}\|_p = \sqrt[p]{\sum_{d=1}^{M} |x_{id} - x_{jd}|^p}$$

– le cosinus :

$$Cos(\overrightarrow{x_i}, \overrightarrow{x_j}) = \frac{\overrightarrow{x_i}.\overrightarrow{x_j}}{||\overrightarrow{x_i}||.||\overrightarrow{x_j}||} = \frac{\sum_{d=1}^{M}(x_{id} \times x_{jd})}{\sqrt{\sum_{d=1}^{M} x_{id}^2 \sum_{d=1}^{M} x_{jd}^2}}$$

– la corrélation de Pearson, cosinus de l'écart à la moyenne :

$$Pearson(\overrightarrow{x_i}, \overrightarrow{x_j}) = \frac{\sum_{d=1}^{M}(x_{id} - \overline{x_i})(x_{jd} - \overline{x_j})}{\sqrt{\sum_{d=1}^{M}(x_{id} - \overline{x_i})^2 \sum_{d=1}^{M}(x_{jd} - \overline{x_j})^2}}$$

avec $\overline{x_i}$ la moyenne des valeurs de l'objet $\overrightarrow{x_i}$ sur l'ensemble des attributs :

$$\overline{x_i} = \frac{\sum_{d=1}^{M} x_{id}}{M}$$

Afin d'éviter que les attributs dont l'intervalle de définition des objets est plus large n'aient plus d'importance que les autres, une phase préalable de normalisation des données est souvent conduite. Typiquement, les valeurs des objets sur les attributs numériques sont normalisés entre 0 et 1. Cette phase sera cependant évitée si on se place dans le cadre d'une application où l'espace de description a un sens géométrique : l'espace euclidien dans le cadre de la segmentation de bases de données spatiales par exemple.

Dans le cas de données décrites par des attributs catégoriels, la similarité entre deux objets est généralement fonction du nombre d'attributs pour lesquels ils partagent la même catégorie. Par exemple, la similarité de Jaccard se calcule ainsi :

$$DJ(\overrightarrow{x_i}, \overrightarrow{x_j}) = \sum_{d=1}^{M} \delta(x_{id}, x_{jd})$$

$$\text{avec } \delta(x_{id}, x_{jd}) = \begin{cases} 0 & \text{si } x_{id} = x_{jd} \\ 1 & \text{si } x_{id} \neq x_{jd} \end{cases}$$

Une autre solution consiste à prendre également en compte les fréquences d'apparition de chaque catégorie sur l'ensemble de la base de données :

$$dist_{\chi^2}(\overrightarrow{x_i}, \overrightarrow{x_j}) = \sum_{d=1}^{M} \frac{N_{x_{id}} + N_{x_{jd}}}{N_{x_{id}} \times N_{x_{jd}}} \times \delta(x_{id}, x_{jd})$$

où $N_{x_{id}} = |\{j \in D | x_{jd} = x_{id}\}|$ représente le nombre total d'objets de la base pour lesquels la catégorie sur l'attribut d est x_{id}.

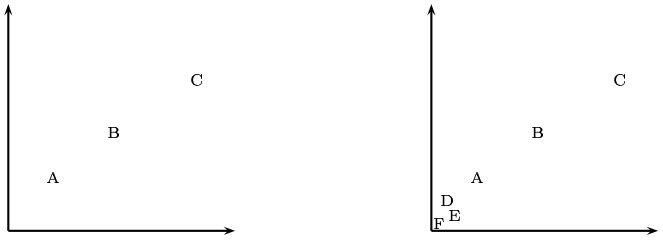
Dans [Guha et al., 2000], un seuil est utilisé pour définir le voisinage entre objets, puis c'est le nombre de voisins en commun entre deux objets, appelés *liens*, qui définit leur similarité.

Dans le même esprit, la *Mutual Neighbor Distance* MND prend en compte l'effet des objets voisins pour définir la similarité entre objets. La figure A.1 illustre alors l'effet

de l'utilisation d'une telle mesure, les objets A et B étant initialement plus similaires que B et C, et inversement après ajout d'autres objets proches de A.

$$MND(\overrightarrow{x_i}, \overrightarrow{x_j}) = NN(\overrightarrow{x_i}, \overrightarrow{x_j}) + NN(\overrightarrow{x_j}, \overrightarrow{x_i})$$

avec $NN(\overrightarrow{x_i}, \overrightarrow{x_j})$ le nombre de voisins de $\overrightarrow{x_j}$ par rapport à $\overrightarrow{x_i}$.



(a) MND(A,B) = 2 < 3 = MND(B,C). (b) MND(A,B) = 5 > 3 = MND(B,C).

FIG. A.1 – Illustration de l'intérêt de la *Mutual Neighbor Distance*.

Enfin, il est également possible de prendre en compte les connaissances d'un expert du domaine dans le calcul de distance entre objets. Celui-ci affecte alors un poids à chaque attribut en fonction de l'importance de cet attribut pour le problème considéré.

A.1.2 Cohésion interne d'un cluster

Étant donnée une mesure de distance entre objets nommée *dist* et choisie parmi celles proposées précédemment ou d'autres, la *cohésion interne* des clusters peut être définie, les deux mesures les plus utilisées dans ce cadre étant les suivantes :

1. le radius, distance moyenne entre membres du cluster et son centroïde, à minimiser pour maximiser la similarité des observations à l'intérieur du cluster :

$$Radius(C_k) = \frac{\sum_{i \in D_k} dist(\overrightarrow{x_i}, \overrightarrow{\mu_k})}{N_k}$$

2. et le diamètre, distance moyenne entre paires de membres du cluster, également à minimiser :

$$Diam(C_k) = \frac{\sum_{i \in D_k} \sum_{j \in D_k} dist(\overrightarrow{x_i}, \overrightarrow{x_j})}{N_k \times (N_k - 1)}$$

A.1.3 Isolation externe d'un cluster

Une autre notion de similarité importante en clustering est celle qui est définie entre différents clusters, qui doit être minimisée pour une bonne *isolation externe* des clusters.

Une première possibilité pour mesurer la similarité entre deux clusters est de calculer la distance entre les objets qui les représentent. Selon la méthode, il s'agira alors de maximiser la distance entre centroïdes, ou bien celle qui sépare les objets les plus proches de chaque cluster.

Cependant, observant qu'en assignant aux $K-1$ premiers clusters (pour K le nombre de clusters recherchés) les $K-1$ objets les moins similaires avec le reste de la base, puis au dernier cluster le reste des objets de la base, la somme pondérée des distances entre centroïdes est maximisée, une autre solution proposée dans [Zhao and Karypis, 2002] pour pallier à ce problème consiste à maximiser la somme des distances entre chaque cluster et le centroïde de l'ensemble des objets de la base :

$$\overrightarrow{x_0} = \frac{\sum_{i=1}^{N} \overrightarrow{x_i}}{N}$$

Trois autres mesures peuvent également être utilisées dans ce cadre :
- la distance moyenne inter-clusters :

$$Inter(k1,k2) = \frac{\sum_{i \in D_{k1}} \sum_{j \in D_{k2}} dist(\overrightarrow{x_i}, \overrightarrow{x_j})}{N_{k1} \times N_{k2}}$$

- la distance moyenne intra-clusters :

$$Intra(k1,k2) = \frac{\sum_{i \in D_{k1} \cup D_{k2}} \sum_{j \in D_{k1} \cup D_{k2}} dist(\overrightarrow{x_i}, \overrightarrow{x_j})}{(N_{k1}+N_{k2})(N_{k1}+N_{k2}-1)} = Diam(C_{k1} \cup C_{k2})$$

- ou la variance globale entre clusters :

$$Var(k1,k2) = \sum_{l \in D_{k1} \cup D_{k2}} dist(\overrightarrow{x_l}, \overrightarrow{\mu_{k1,k2}}) - \sum_{i \in D_{k1}} dist(\overrightarrow{x_i}, \overrightarrow{\mu_{k1}}) - \sum_{j \in D_{k2}} dist(\overrightarrow{x_j}, \overrightarrow{\mu_{k2}})$$

A.1.4 Mesures internes de la qualité d'un clustering

Les principales mesures utilisées ensuite pour évaluer la *qualité interne* d'un clustering P sont les suivantes :

1. l'erreur quadratique :

$$E^2(P) = \sum_{k=1}^{K} \sum_{i \in D_k} dist(\overrightarrow{x_i}, \overrightarrow{\mu_k})$$

qui peut éventuellement être pondérée par les probabilités P_{ik} d'appartenance des objets $\overrightarrow{x_i}$ aux clusters C_k dans le cas du *soft clustering* :

$$E_p^2(P) = \sum_{k=1}^{K} \sum_{i=1}^{N} P_{ik} \times dist(\overrightarrow{x_i}, \overrightarrow{\mu_k})$$

avec dans ce cas $\overrightarrow{\mu_k} = \sum_{i=1}^{N} P_{ik} \times \overrightarrow{x_i}$

2. la moyenne pondérée du carré des radius de clusters :

$$Q1(P) = \sum_{k=1}^{K} \frac{N_k}{N} \times Radius(C_k)^2$$

3. la moyenne pondérée du carré des diamètres de clusters :

$$Q2(P) = \frac{\sum_{k=1}^{K} N_k \times (N_k - 1) \times Diam(C_k)^2}{\sum_{k=1}^{K} N_k \times (N_k - 1)}$$

4. la mesure hybride mixant cohésion interne et isolation externe proposée dans [Zhao and Karypis, 2002], où $\overrightarrow{x_0}$ dénote le centroïde de l'ensemble des objets de la base :

$$H(P) = \frac{\sum_{k=1}^{K} \sum_{i \in D_k} dist(\overrightarrow{x_i}, \overrightarrow{\mu_k})}{\sum_{k=1}^{K} N_k \times dist(\overrightarrow{\mu_k}, \overrightarrow{x_0})}$$

5. ou la mesure nommée *Partition Utility* basée sur la *Category Utility* de Fisher [Fisher, 1987] :

$$PU(P) = \sum_{k=1}^{K} \frac{N_k}{N} \times CU(C_k)$$

$$CU(C_k) = P(C_k) \sum_{d=1}^{M} \sum_{\{m \in Modalites_d\}} \Big[P(x_{id} = m | i \in D_k)^2 - P(x_{id} = m | i \in D)^2 \Big]$$

l'idée ici étant de favoriser les clusters qui vont accroître la prédictabilité, c'est à dire ceux pour lesquels on a :

$$P(x_{id} = m | i \in D_k) > P(x_{id} = m | i \in D)$$

cette mesure ayant été adaptée comme suit au cas des dimensions numériques dans [Gennari et al., 1989] :

$$CU(C_k) = P(C_k) \frac{1}{\sqrt{2\pi}} \sum_{d=1}^{M} \left(\frac{1}{\sigma_{kd}} - \frac{1}{\sigma_d} \right)$$

où σ_{kd} est la déviation standard des valeurs des objets appartenant au cluster C_k sur la dimension d, et σ_d la déviation standard des valeurs de l'ensemble des objets de la base sur la dimension d.

A.1.5 Mesures externes de la qualité d'un clustering

En ce qui concerne les mesures de la *qualité externe* d'un clustering, c'est-à-dire dans le cas où l'on souhaite comparer les résultats d'un clustering P à une cible connue a priori P^0, nous avons déjà vu que l'entropie ou la F-mesure présentées dans la section 2.6 peuvent être utilisées.

Dans [Meila, 2003] et [Patrikainen and Meila, 2004], une mesure de *Variation d'Information* est proposée pour comparer deux solutions. Son fondement consiste à évaluer la quantité d'information qui est gagnée/perdue lors du passage d'une solution à l'autre, et inversement. Étant donné le clustering résultat attendu P^0, et la mesure d'entropie

$E(P)$ associée à un clustering P, la mesure de Variation d'Information entre un clustering donné P et le clustering attendu P^0 est la suivante :

$$VI(P, P^0) = E(P) + E(P^0) - 2 \times I(P, P^0)$$

$$I(P, P^0) = \sum_{k=1}^{K} \sum_{l=1}^{K} P(k,l) \log \frac{P(k,l)}{P(k)P(l)}$$

$$P(k) = \frac{N_k}{N} \text{ et } P(k,l) = \frac{|C_k^0 \cap C_l|}{N}$$

A.2 Problématique des larges bases de données

Pour être applicable à des bases de données contenant un grand nombre d'objets ou d'attributs, les méthodes de clustering développées doivent porter une attention particulière à leur complexité. Typiquement, la meilleure solution est de développer des méthodes qui dépendent de façon linéaire du nombre d'objets et d'attributs. D'autres solutions peuvent également être envisagées :

1. minimiser le nombre de parcours de l'ensemble des données ;
2. réduire le nombre de données examinées pendant l'exécution ;
3. réduire la taille de la structure des données fournie à l'algorithme.

Plus concrètement, les solutions les plus souvent utilisées dans ce cadre sont les suivantes [Hinneburg and Keim, 1999] :

1. l'échantillonnage aléatoire permet de sélectionner un sous-ensemble représentatif des données, permettant ainsi de considérer ensuite moins d'objets lors du déroulement de la méthode ;
2. l'utilisation de bornes permet de fixer un nombre maximum d'itérations de la méthode, lorsque celle-ci est capable de fournir une solution à chaque instant, comme c'est le cas pour les méthodes stochastiques ou statistiques ;
3. le partitionnement permet de diviser le problème en plusieurs sous-problèmes de plus petite taille, puis d'utiliser ensuite les résultats de ces sous-problèmes pour résoudre le problème général ;
4. et la transformation des données permet de réduire la taille des données fournies en entrée de la méthode, afin de travailler ensuite sur une base de données moins importante : il s'agit dans ce cas de transformer les données sous une représentation plus compacte, ou bien de se placer dans un espace plus réduit que l'espace original ; ce dernier point sera détaillé dans l'annexe suivante.

A.3 Clustering hiérarchique

A.3.1 COBWEB

COBWEB [Fisher, 1987] est un exemple d'algorithme de clustering hiérarchique descendant. Il est incrémental, ce qui permet de ne scanner les données qu'une seule fois mais le rend dépendant de l'ordre dans lequel ces données sont présentées à l'algorithme. La méthode gérant l'intégration d'un nouvel élément dans la hiérarchie est la suivante :

1. démarrer de la racine de l'arbre ;
2. parmi les quatre opérateurs applicables sur le cluster courant, sélectionner celui qui optimise la mesure CU (Category Utility) définie dans la section A.1.4 :
   (a) intégrer l'élément dans le cluster courant ;
   (b) fusionner deux clusters existants ;
   (c) diviser le cluster courant ;
   (d) ou créer un nouveau cluster pour l'élément ;
3. et recommencer à l'étape 2 en choisissant comme nouveau cluster courant celui qui a été la cible de l'opération précédente, jusqu'à arriver à une feuille de l'arbre.

A.3.2 CURE

CURE [Guha et al., 1998], quant à lui, est un exemple d'algorithme de clustering hiérarchique ascendant. Dans cette méthode, les clusters sont représentés par un ensemble d'objets distants qui les composent, rapprochés de leur centroïde selon un facteur α. Pour décider quels clusters doivent être fusionnés, il choisit les clusters dont la paire d'objets représentatifs est la plus proche.

Et pour faire face aux larges bases de données, CURE utilise l'échantillonnage aléatoire et le partitionnement, c'est-à-dire qu'il effectue un clustering sur différentes sous-parties de l'espace, puis combine les résultats obtenus pour effectuer le clustering global.

A.4 Clustering K-médoïdes

A.4.1 CLARANS

Pour faire face aux larges bases de données, les auteurs de [Ng and Han, 1994], dans leur système appelé CLARANS, proposent d'utiliser une recherche stochastique basée sur différents paramètres permettant de borner le nombre d'itérations de la méthode, ainsi que sur l'échantillonage aléatoire. Étant donné le nombre K de clusters recherchés, une solution consiste en un ensemble de K médoïdes, objets représentatifs des clusters, auxquels sont associés l'ensemble des objets en fonction de leur proximité avec ces médoïdes. Les étapes principales de la méthode sont les suivantes :

1. sélectionner un échantillon représentatif des données ;
2. itérer un certain nombre fixé de fois :

(a) choix d'une solution aléatoire : un ensemble de K médoïdes ;

(b) itérer un certain nombre fixé de fois :
- choix d'une solution voisine de la solution courante, par modification aléatoire de l'un des médoïdes de la solution ;
- conservation du voisin comme nouvelle solution courante si l'inertie globale de la partition est inférieure à celle de la solution précédente ;

(c) stocker la solution optimale locale trouvée ;

3. et retourner la meilleure des solutions optimales locales trouvées.

A.4.2 BIRCH

Afin d'éviter de scanner plusieurs fois l'ensemble des données, une méthode d'amélioration du stockage des données a été proposée dans [Zhang et al., 1997]. Pour cela, l'idée principale de la méthode, appelée BIRCH, est de condenser initialement les données et de les stocker dans un arbre appelé *CF-Tree*, en considérant collectivement les objets proches.

Ainsi, les données fournies en entrée de l'algorithme ne seront scannées qu'une seule fois, au sein même de l'algorithme, mais aussi si l'on souhaite tester plusieurs paramètres, et l'algorithme pourra être exécuté sur une base de données de beaucoup plus petite taille et de meilleure qualité en ce qui concerne l'ordre dans lequel les données seront fournies en entrée de l'algorithme.

Un élément CF (*Clustering Feature*) du *CF-Tree* est composé de trois paramètres : $(n, \overrightarrow{LS}, SS)$, avec n le nombre d'objets le composant, $\overrightarrow{LS} = \sum_{i=1}^{n} \overrightarrow{x_i}$ la somme linéaire de ces n objets, et $SS = \sum_{i=1}^{n} \overrightarrow{x_i}^2$ leur somme au carré. Cette représentation permet en outre le *théorème de CF additivité* suivant :

$$CF_1 + CF_2 = (n_1 + n_2, \overrightarrow{LS_1} + \overrightarrow{LS_2}, SS_1 + SS_2)$$

Un *CF-Tree* est un arbre basé sur deux paramètres permettant de réguler sa taille : un facteur de branchement (B pour les nœuds, et L pour les feuilles), et un seuil T.

Chaque nœud contient au plus B entrées de la forme [CF_i,$child_i$], où $i \in [1, B]$, $child_i$ est un pointeur vers le i^{eme} nœud fils, et CF_i est l'entrée CF du sous-cluster représenté par ce fils. Un nœud de l'arbre représente donc un sous-cluster composé de tous les sous-clusters représentés par ses entrées.

De même, chaque feuille contient au plus L entrées, et chaque entrée est un CF. De plus, à chaque feuille sont associés deux pointeurs utilisés pour chaîner toutes les feuilles ensemble pour un parcours efficace. Une feuille représente donc un sous-cluster composé de tous les sous-clusters représentés par ses entrées. Enfin, toutes les entrées dans une feuille doivent satisfaire une *condition de seuil* : le diamètre (ou le radius) de chaque entrée dans une feuille doit être inférieur à T.

L'algorithme CLARANS est ensuite utilisé, puis chacun des objets initiaux est finalement redistribué par rapport aux K médoïdes trouvés.

A.5 Clustering stochastique

L'avantage des algorithmes de clustering stochastiques face aux larges bases de données est qu'ils sont capables de fournir une solution au problème à n'importe quel moment de l'exécution de l'algorithme, de telle sorte que l'utilisateur peut choisir d'interrompre la recherche quand il le souhaite.

En pratique, les méthodes mises en œuvre sont souvent construites de manière à trouver un bon compromis entre l'exploration nécessaire d'une partie importante de l'espace des solutions, et l'exploitation des solutions optimales identifiées, au sens de l'optimisation d'une fonction critère cible choisie, par exemple parmi celles proposées dans la section A.1.4.

A.5.1 Algorithmes génétiques

Basé sur le paradigme de l'évolution génétique des individus, le clustering par *algorithmes génétiques* consiste à effectuer une succession de sélections et de mutations d'un ensemble de solutions, afin d'améliorer celui-ci au fur et à mesure de l'exécution de la méthode.

Étant données deux solutions *parents* au problème, d'autres solutions *enfants* sont générées à partir des parents, et une partie de celles qui optimisent la fonction cible sont sélectionnées, et utilisées à leur tour en tant que parents à l'étape suivante de la méthode. Certaines solutions non optimales sont également conservées afin de favoriser l'exploration de l'espace des solutions et éviter de rester enfermé dans des optima locaux de la fonction.

La mutation la plus utilisée dans ce cadre est celle utilisant le *croisement* entre individus. Cette technique, appliquée à l'exemple de la figure 1.1, est illustrée par la figure A.2. Une solution consiste en un ensemble d'associations entre les objets et les clusters, représenté par un vecteur de couples d'identifiants des objets et clusters considérés. Une *zone de coupure* dans les vecteurs solutions est choisie de façon aléatoire, et deux enfants sont générés à partir de deux parents, l'un avec la première partie de la solution du premier parent et la deuxième partie de l'autre parent, et inversement pour le deuxième enfant.

A.5.2 Recherche Tabou

Un autre méthode stochastique de clustering est celle basée sur la *recherche Tabou*. Une solution au problème est sélectionnée aléatoirement dans l'espace des solutions possibles. Puis à chaque itération de la *recherche Tabou*, une nouvelle solution voisine de la précédente est considérée, c'est-à-dire qu'une ou plusieurs associations entre objets et clusters sont modifiées. Si cette nouvelle solution est meilleure que la solution précédente, alors elle est conservée pour l'itération suivante. Sinon, un autre voisin est envisagé et évalué. Si aucun voisin n'est considéré comme meilleur que la solution courante, alors celle-ci est conservée en tant que minimum local, et la méthode est itérée avec une autre solution initiale sélectionnée aléatoirement. Au fur et à mesure de l'exploration des solutions possibles, une partie de celles qui ont déjà été rencontrées sont stockées

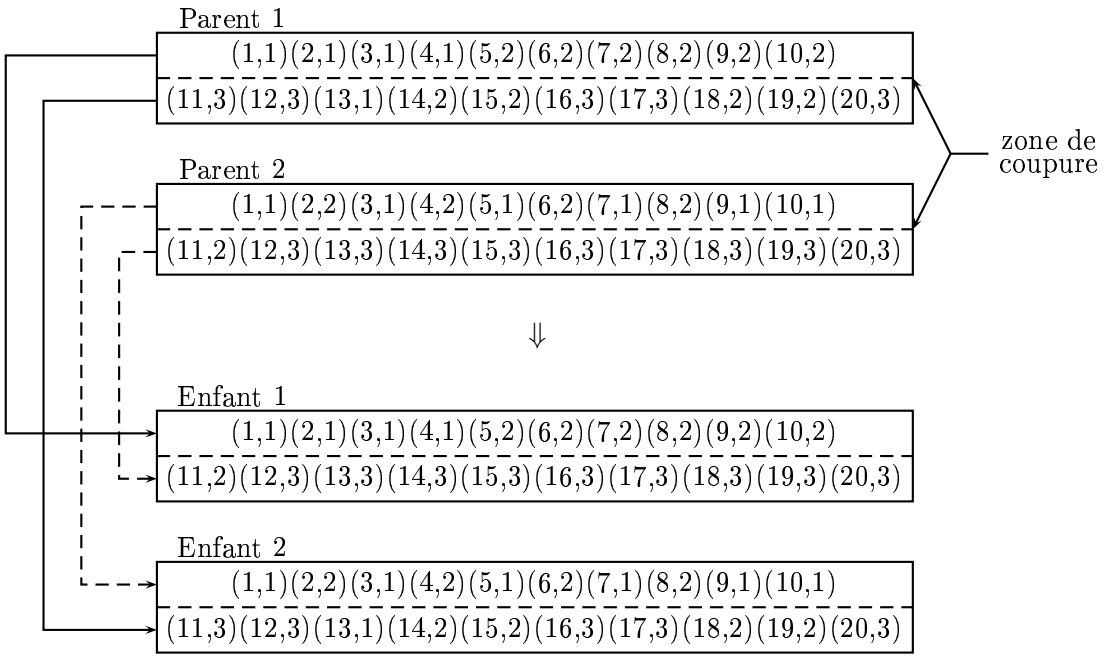


FIG. A.2 – La mutation par croisement génétique.

dans une *liste Tabou* utilisée pour guider la recherche et éviter de considérer plusieurs fois une même solution.

A.5.3 Recuit simulé

Enfin, une autre méthode de clustering stochastique peut être envisagée : le clustering par *recuit simulé*. Dans ce cas, à chaque itération de la méthode, de nouvelles solutions sont envisagées et conservées si elles sont meilleures que les précédentes. Un paramètre, appelé la *température*, contrôle les nouvelles solutions proposées pour l'itération suivante : plus la température est faible, plus les solutions suivantes considérées sont proches des solutions précédentes. Initialement la valeur de la température est haute, les solutions suivantes considérées éloignées des précédentes et donc l'exploration de l'espace des solutions est privilégiée. Puis au fur et à mesure de l'exécution de l'algorithme, la température est diminuée et les solutions envisagées sont de plus en plus proches les unes des autres pour favoriser l'exploitation des solutions optimales courantes sélectionnées.

A.6 Clustering basé sur la densité

Dans le cas du clustering basé sur la densité, un avantage important est le fait qu'il est capable de cibler des clusters de forme quelconque. Cependant, la méthode de base DBSCAN présentée dans la section 3.5 souffre de sa dépendance vis-à-vis des paramètres en entrée *Eps* et *MinPts* contrôlant la notion de densité du voisinage d'un objet. C'est pourquoi plusieurs méthodes ont été proposées pour faire face à cette problématique.

A.6.1 OPTICS

Afin de permettre d'identifier des clusters ayant des densités très différentes, comme dans le cas de la figure 2.3, il a été proposé dans [Ankerst et al., 1999] d'ajouter la fonctionnalité de faire varier la valeur du paramètre *Eps*. Initialement basse, cette valeur est peu à peu augmentée, et des clusters de moins en moins denses sont recherchés. L'autre avantage de cette technique est que l'utilisateur n'a plus à gérer la valeur de ce paramètre.

A.6.2 DBCLASD

Dans [Xu et al., 1998], un autre critère d'acceptation de l'intégration de nouveaux objets dans les clusters a été proposé afin de se débarrasser des deux paramètres *Eps* et *MinPts*. La méthode se base pour cela sur l'hypothèse que les objets appartenant à un même cluster sont distribués de façon uniforme à l'intérieur du cluster. Ainsi, lorsqu'un objet est candidat à l'intégration dans un cluster, la technique consiste à vérifier que les distances avec ses plus proches voisins suive la distribution des distances courante entre membres du cluster.

Pour initialiser ces distributions à l'intérieur des clusters, un nombre minimum d'objets associés aux clusters est nécessaire. La méthode ayant été créée dans le but de faire face à des bases de données importantes, l'initialisation proposée consiste à considérer les trente plus proches voisins d'un objet comme solution initiale. Par ailleurs, cette méthode est dépendante de l'ordre dans lequel les objets sont présentés. Les tentatives d'intégration des objets aux clusters sont donc itérées plusieurs fois, et un post-traitement peut être utilisé pour permettre aux objets de changer d'affectation de cluster.

A.6.3 DENCLUE

Dans [Hinneburg and Keim, 1998], une fonction de distribution de la densité est utilisée pour décider si des objets font partie d'un même cluster. Une fonction influence, par exemple gaussienne, est d'abord définie pour mesurer l'impact d'un objet dans son voisinage :

$$f(\overrightarrow{x_i}, \overrightarrow{x_j}) = e^{-\frac{dist(\overrightarrow{x_i}, \overrightarrow{x_j})^2}{2\sigma^2}}$$

La fonction de densité globale est alors estimée par la somme des fonctions influence de tous les objets :

$$f^D(\overrightarrow{x_i}) = \sum_{j=1}^{N} e^{-\frac{dist(\overrightarrow{x_i}, \overrightarrow{x_j})^2}{2\sigma^2}}$$

On appelle *attracteurs* les objets pour lesquels la fonction de densité globale est localement maximale. Chaque objet est associé à son *attracteur* le plus approprié en utilisant une méthode de gradient. Enfin, deux *attracteurs*, ainsi que les objets qui leur sont associés, sont assignés au même cluster s'il existe un chemin dans l'espace de description des objets $P \in S$ qui les relie tel que la densité globale de tout objet $\overrightarrow{x_i}$ présent sur ce chemin soit supérieure à un seuil fixé ξ :

$$f^D(\overrightarrow{x_i}) \geq \xi \ \forall \ \overrightarrow{x_i} \in P$$

Deux paramètres sont donc influants pour cette méthode :

1. σ, qui détermine l'influence d'un point dans son voisinage ;
2. et ξ, qui détermine le niveau de pertinence des *attracteurs*.

Une méthode est donc finalement proposée pour fixer les valeurs de ces deux paramètres influant du système σ et ξ.

Annexe B

Prise en compte du contexte

B.1 Modification de l'espace de description

Afin de prendre en compte le fait que certains attributs décrivant les données ont plus ont moins d'importance pour la classification, une première possibilité consiste à utiliser une méthode de sélection, d'extraction ou de pondération d'attributs en prétraitement de l'apprentissage.

La sélection d'attributs est le processus qui consiste à identifier le sous-ensemble des attributs le plus efficace à utiliser pour le clustering. L'extraction d'attributs correspond à l'utilisation d'une ou plusieurs transformations des attributs initiaux pour produire de nouveaux attributs pertinents. Et la pondération d'attributs consiste à donner un poids plus important à certains attributs considérés comme plus pertinents pour le clustering.

B.1.1 Sélection d'attributs

Il existe trois grandes techniques pour sélectionner le sous-ensemble des attributs le plus approprié pour le clustering : exhaustive, aléatoire ou heuristique. Étant donné un critère d'intérêt d'un sous-ensemble d'attributs pour effectuer le clustering d'un ensemble d'objets, il s'agit dans le premier cas de considérer l'ensemble des sous-ensembles d'attributs possibles et de sélectionner celui qui optimise ce critère.

Une telle approche n'est cependant pas utilisable en pratique à cause de sa complexité exponentielle $O(2^M)$ en fonction du nombre de dimensions M. Une approche aléatoire consiste à sélectionner aléatoirement puis évaluer différents sous-ensembles d'attributs, son avantage étant alors d'être capable de fournir un résultat à n'importe quel moment de son exécution.

Enfin, des heuristiques souvent utilisées en sélection d'attributs consistent à effectuer une sélection ascendante ou descendante. En sélection ascendante, l'attribut considéré comme le plus pertinent selon le critère utilisé et parmi l'ensemble des attributs de la base est d'abord sélectionné, puis le second attribut le plus pertinent est sélectionné, et ainsi de suite. L'opération inverse est exécutée en sélection descendante : l'ensemble des dimensions de description est initialement considéré, puis à chaque étape, la dimension évaluée comme la moins pertinente est supprimée.

Dans [Dash et al., 2002], le critère de la pertinence d'un sous-ensemble d'attributs pour le clustering est basé sur l'observation qu'un ensemble de données contenant des clusters a un histogramme des distances entre objets très différent par rapport à celui d'un ensemble de données ne contenant pas de clusters. En effet, si un ensemble de données peut être partitionné, alors la majorité des distances intra-clusters sera bien inférieure à la majorité des distances inter-clusters. Au contraire, si les données sont uniformément réparties, alors les distances entre objets le seront également. La mesure proposée est dérivée de la mesure d'entropie, qui représente typiquement le taux de désordre d'une configuration, D_{ij} dénotant la distance normalisée entre les objets $\overrightarrow{x_i}$ et $\overrightarrow{x_j}$ dans le sous-espace considéré :

$$E = -\sum_{i=1}^{N}\sum_{j=1}^{N}[D_{ij} \times \log D_{ij} + (1 - D_{ij}) \times \log(1 - D_{ij})]$$

Une autre mesure basée sur la *Category Utility* présentée dans la section A.1.4 a également été proposée dans [Talavera, 2000] pour évaluer la pertinence d'attributs catégoriels, l'hypothèse sous-jacente à cette mesure étant de considérer que si une dimension n'est pas hautement corrélée aux autres, alors cette dimension ne jouera pas un rôle important dans le processus de clustering, et peut dès lors être considérée comme peu pertinente :

$$\frac{\sum_{l=1}^{M}\sum_{m\in Modalites_l} P(x_{il}=m)\sum_{m2\in Modalites_d}[P(x_{id}=m2|x_{il}=m)^2 - P(x_{id}=m2)^2]}{M-1}$$

B.1.2 Extraction d'attributs

La technique d'extraction d'attributs la plus utilisée est sûrement l'*Analyse en Composantes Principales* (ACP) [Jolliffe, 1986]. Cette technique consiste à définir un ensemble de m nouvelles variables, combinaison linéaire des M variables de l'espace initial, qui feraient perdre le moins d'information possible. Ces m variables sont appelées *composantes principales* et les axes qu'elles déterminent *axes principaux*.

Il s'agit pour cela d'analyser la structure de la matrice variance-covariance, qui rend compte de la variabilité, de la dispersion des données, l'objectif de l'ACP étant de décrire à l'aide d'un minimum de composantes un maximum de cette variabilité. Cette technique permet alors de réduire efficacement la taille des données considérées. Elle peut aussi être utilisée afin de visualiser les données en deux ou trois dimensions.

La première étape de la méthode consiste à standardiser les données, c'est-à-dire les centrer et les réduire. Ensuite, il s'agit de constituer la matrice de corrélation entre les variables, puis de trouver les vecteurs propres de cette matrice, ainsi que leur valeur propre associée. Les vecteurs propres donnent alors les axes factoriels. Ils sont déterminés de façon à rendre compte le mieux possible de la dispersion des données présentes dans la matrice. Les valeurs propres, quant à elles, sont proportionnelles à la variance associée à ces axes. La dernière étape de la méthode consiste finalement à calculer les coordonnées

des individus sur les nouveaux axes sélectionnés, c'est-à-dire les m vecteurs propres dont les valeurs propres associées sont maximales.

La figure B.1 montre sur un exemple les deux axes principaux $\overrightarrow{p_1}$ et $\overrightarrow{p_2}$ calculés à partir d'un ensemble d'objets définis sur deux dimensions.

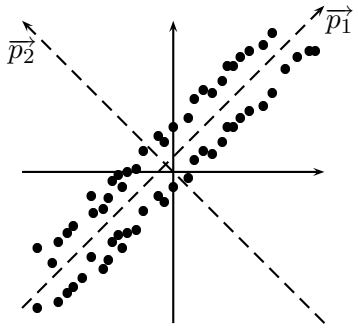


FIG. B.1 – Exemple d'Analyse en Composantes Principales

B.1.3 Pondération d'attributs

Dans [Friedman and Meulman, 2004], les auteurs proposent d'assigner à chaque objet un poids sur les dimensions de l'espace. Basée sur l'utilisation des plus proches voisins, la technique est la suivante :

- démarrer avec toutes les dimensions également pondérées pour chaque objet ;
- cibler les k plus proches voisins de chaque objet ;
- plus la dispersion du groupe des k plus proches voisins sur une dimension est faible, plus le poids assigné à cette dimension est important ;
- ces poids sont ensuite utilisés pour calculer les nouvelles distances, qui sont réutilisées pour cibler les k *nouveaux* plus proches voisins ;
- et le processus est ainsi répété jusqu'à ce que les poids se stabilisent.

L'algorithme fournit alors en sortie une matrice de distances entre objets, qui peut être utilisée par n'importe quel algorithme de clustering, et permettra ainsi de refléter le fait que les dimensions à prendre en compte pour les calculs de distance entre objets peuvent varier d'un couple à l'autre.

B.2 Subspace clustering ascendant

B.2.1 CLIQUE

La première méthode proposée dans le cadre du subspace clustering est appelée CLIQUE [Agrawal et al., 1998]. Basée sur l'utilisation de grilles, son principe général est de rechercher les ensembles de cellules denses connectées dans des sous-espaces de l'espace original. Comme toute méthode basée sur les grilles, elle nécessite donc la donnée de deux paramètres utilisateurs : ξ est un entier utilisé pour déterminer la

taille de la grille à utiliser, obtenue en partitionnant chaque dimension de l'espace de description des objets en ξ intervalles de même longueur, et τ représente le pourcentage minimum d'objets qui doivent être contenus dans une cellule de la grille pour qu'elle soit considérée comme dense.

La méthode employée par CLIQUE se décompose en trois phases :

1. cibler les sous-espaces dans lesquels se trouvent des clusters denses ;
2. identifier les clusters dans ces sous-espaces ;
3. générer une description minimale des clusters trouvés.

Concernant la première phase, l'algorithme utilise la propriété de monotonie suivante : *si un ensemble d'objets C_k est un cluster dans un sous-espace à m dimensions, alors C_k fait également partie d'un cluster dans toute projection à $(m-1)$ dimensions de ce sous-espace.*

Basée sur ce principe, une approche ascendante est donc utilisée pour cibler les sous-espaces dans lesquels se trouvent des clusters denses. On démarre en projetant les objets sur chaque dimension de l'espace. Pour sélectionner les sous-espaces pertinents à considérer (ceux étant susceptibles de contenir des clusters denses), une technique possible est de considérer ceux qui maximisent la couverture de l'espace des objets par les cellules denses (mesure qui sera faible si les objets sont uniformément distribués dans le sous-espace, et inversement), en se basant sur le principe MDL (*Minimum Description Length*) pour décider de la zone de coupure entre sous-espaces conservés et sous-espaces supprimés.

Puis, pour passer des sous-espaces candidats S_{m-1} à $(m-1)$ dimensions aux sous-espaces candidats S_m à m dimensions, on combine deux sous-espaces de S_{m-1} à $(m-1)$ dimensions si ceux-ci ont leur $(m-2)$ premières dimensions en commun, et si tout sous-espace à $(m-1)$ dimensions inclus dans ce nouveau sous-espace à m dimensions est déjà inclus dans S_{m-1}.

Ensuite, lorsque les sous-espaces ont été trouvés et qu'il n'y a plus d'autre sous-espace candidat, l'étape suivante de la méthode consiste à cibler les clusters dans ces sous-espaces, c'est-à-dire à identifier les ensembles de cellules denses connectées en utilisant un clustering basé sur les grilles, puis de donner une description minimale de ces clusters, sous forme normale disjonctive.

Pour cibler les clusters dans un sous-espace donné, la méthode choisie est récurrente et admet deux paramètres : une cellule u et un numéro de cluster n. La méthode est démarrée avec une cellule dense sélectionnée aléatoirement et un numéro de cluster initial de 1. Puis la procédure récurrente, notée $dfs(u,n)$, est la suivante : sur chaque dimension de l'espace, et pour chacun des deux voisins v (droite et gauche) de la cellule u, si v est dense et qu'aucun cluster ne lui est encore assigné, alors l'assigner au cluster courant n et lancer $dfs(v,n)$. Puis s'il reste encore des cellules non assignées à un cluster, en sélectionner une aléatoirement, et relancer la méthode après avoir incrémenté n.

Enfin, la dernière phase de description des clusters se décompose en deux étapes :

1. effectuer une couverture de chaque cluster par des régions maximales ;
2. effectuer une sélection parmi ces régions pour obtenir une couverture minimale.

La première phase se fait par un algorithme glouton :
- démarrer avec une cellule sélectionnée aléatoirement ;
- sélectionner une dimension aléatoirement et élargir la région courante sur la droite ou la gauche si la nouvelle région ne couvre pas de cellule non dense, jusqu'à ce que toutes les dimensions soient envisagées ;
- et ainsi de suite jusqu'à ce que toutes les cellules soient couvertes.

Puis finalement, pour obtenir une couverture minimale, on supprime les régions redondantes en commençant par celles qui couvrent le moins de cellules.

B.2.2 ENCLUS

Dans [Cheng et al., 1999], les auteurs partent de la remarque que la mesure de sélection des sous-espaces pertinents dans CLIQUE (la couverture de l'espace des objets par les cellules denses) n'est pas suffisante. Ils proposent de tenir compte également de la densité elle-même des cellules. En effet, si nous considérons le cas de la figure B.2, qui montre la fonction de densité de probabilité d'une variable aléatoire X, on remarque que la couverture de l'espace des objets par les cellules denses est la même dans les deux cas, mais pas la densité elle-même, qui est plus élevée dans le second cas. Or cette notion de forte densité n'est pas prise en compte dans CLIQUE.

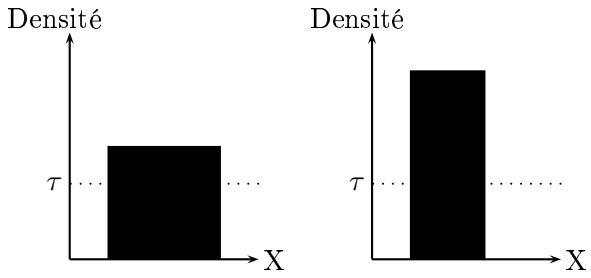


FIG. B.2 – Deux sous-espaces de couverture identique mais de densité différente.

Pour affiner cette sélection des sous-espaces pertinents à considérer, la proposition consiste donc à utiliser une autre mesure : l'entropie, la motivation étant qu'un sous-espace contenant des clusters a typiquement une entropie inférieure aux sous-espaces ne contenant pas de cluster.

Soit χ l'ensemble des cellules considérées et $p(x)$ la fonction de densité de probabilité de la variable aléatoire X. L'entropie $H(X)$ est alors définie par :

$$H(X) = -\sum_{x \in \chi} p(x) \times \log p(x)$$

Cette mesure est maximale lorsque les objets sont uniformément distribués dans l'espace de description, et inversement.

De plus, l'entropie est également utilisée pour calculer la corrélation entre les dimensions du sous-espace concerné. Cela permet donc de détecter si les sous-espaces

considérés ne contiennent pas des dimensions redondantes. Par exemple, dans le cas de la figure B.3, considérer les deux dimensions X et Y est nécessaire pour identifier les deux clusters dans le premier cas, alors qu'une seule dimension suffit pour cibler l'unique cluster existant dans le second cas.

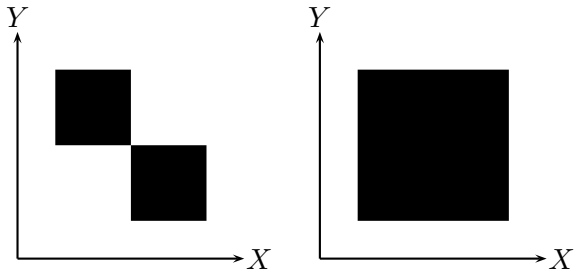


FIG. B.3 – Détection de la présence de corrélations entre dimensions.

Finalement , la nouvelle mesure introduite à maximiser est la suivante :

$$interest(\{X_1, ..., X_n\}) = \sum_{i=1}^{n} H(X_i) - H(X_1, ..., X_n)$$

$$H(X_1, ..., X_n) = - \sum_{x_1 \in \chi_1} ... \sum_{x_n \in \chi_n} p(x_1, ..., x_n) \times \log p(x_1, ..., x_n)$$

l'idée étant que si $H(X_1, ..., X_n) = \sum_{i=1}^{n} H(X_i)$, c'est que les dimensions sont indépendantes.

B.2.3 MAFIA

Constatant que l'utilisation de grilles fixes dans CLIQUE conduit à plusieurs problèmes au niveau du temps d'exécution et de la précision des résultats de l'algorithme, il a été proposé dans [Nagesh et al., 1999] d'adapter le partitionnement des sous-espaces en fonction de la répartition des objets. Le nombre de cellules générées est alors souvent réduit, et la phase de description minimale des clusters est menée en même temps que le partitionnement, et non pas a posteriori. La construction d'une telle grille adaptative sur chaque dimension suit les étapes suivantes :

1. créer l'histogramme de la répartition des objets sur la dimension ;
2. diviser la dimension en fenêtres de très petite taille (5%) ;
3. à chaque cellule est associée le maximum des valeurs de l'histogramme qui la composent ;
4. enfin, deux cellules dont les valeurs sont proches ($\pm$20%) sont fusionnées.

Enfin, lorsqu'il s'agit de passer de sous-espaces à $(m-1)$ dimensions à des sous-espaces à m dimensions, la méthode ne considère pas uniquement les combinaisons de sous-espaces à $(m-1)$ dimensions qui ont leurs $(m-2)$ *premières* dimensions en

commun, mais toute combinaison de sous-espaces à $(m-1)$ dimensions qui ont $(m-2)$ dimensions en commun. Un traitement parallèle est également proposé afin d'accélérer la méthode.

B.2.4 SUBCLU

La méthode présentée dans [Kailing et al., 2004] étend celle de DBSCAN présentée dans la section 3.5 au cas du subspace clustering, en utilisant une aproche ascendante identique aux méthodes précédentes : il commence par partitionner les données sur chaque dimension indépendamment, puis utilise les partitions effectuées dans les sous-espaces à m dimensions pour passer à des partitions dans des sous-espaces à $(m+1)$ dimensions. Mais au lieu de partitionner le sous-espace étudié à l'aide d'une grille, il utilise DBSCAN pour identifier les clusters présents dans le sous-espace concerné.

Pour décider d'investiguer ou non un sous-espace donné S_m à m dimensions, il vérifie si tous les sous-espaces $S_{m-1} \subset S_m$ à $(m-1)$ dimensions inclus dans S_m contiennent des clusters. Car si un sous-espace $S_{m-1} \subset S_m$ ne contient pas de cluster, alors S_m ne contiendra pas non plus de cluster, selon le même principe de monotonie utilisé par les méthodes précédentes.

Enfin, pour améliorer les performances de l'algorithme, il est possible d'utiliser les partitions dans les sous-espaces S_{m-1} pour effectuer la partition dans S_m. Et pour choisir le sous-espace $S_{m-1} \subset S_m$ à considérer pour cela, l'heuristique que les auteurs proposent est de sélectionner le sous-espace dans lequel un minimum d'objets a été utilisé pour former les clusters.

B.3 Subspace clustering descendant

B.3.1 PROCLUS

La première méthode de subspace clustering ascendant qui a été présentée dans [Aggarwal et al., 1999] est basée sur une recherche stochastique par K-médoïdes et deux paramètres utilisateurs : le nombre K de clusters recherchés et le nombre moyen L de dimensions nécessaires pour les décrire.

Un *cluster projeté* C_k est défini comme un sous-ensemble D_k des objets associé à un sous-ensemble S_k des dimensions tels que les objets de D_k soient fortement groupés dans le sous-espace des dimensions S_k. La méthode utilisée ensuite se divise en trois phases principales :

1. sélection d'un ensemble initial de K médoïdes ;
2. recherche des médoïdes les plus appropriés aux données et des sous-espaces les plus appropriés aux médoïdes ;
3. raffinement du résultat.

Dans la phase d'initialisation, K médoïdes sont sélectionnés de façon itérative de telle façon que le médoïde choisi soit le plus éloigné possible de l'ensemble des médoïdes sélectionnés jusqu'alors.

La seconde phase consiste à itérer trois étapes pour optimiser l'ensemble des K médoïdes représentant les clusters recherchés, ainsi que les sous-ensembles des dimensions associées à ces médoïdes :

1. considérer une nouvelle solution de K médoïdes ;
2. rechercher les dimensions associées aux médoïdes ;
3. évaluer la partition.

Dans un premier temps, une solution voisine de la solution courante est proposée. Celle-ci est construite en remplaçant les médoïdes auxquels sont associés le moins d'objets par d'autres médoïdes choisis aléatoirement dans l'ensemble des objets. Si une solution est considérée comme plus pertinente que la solution courante, alors celle-ci devient la nouvelle solution courante et la méthode est itérée. Sinon, une autre solution voisine est sélectionnée et évaluée. Le processus se termine lorsque, après un nombre fixé d'itérations successives, aucune autre solution n'est considérée comme plus pertinente que la solution optimale courante, le critère de qualité d'une partition étant basé sur la moyenne pondérée des radius des clusters, et la distance utilisée entre un objet $\overrightarrow{x_i}$ et un cluster C_k tenant compte du sous-espace dans lequel le cluster est projeté :

$$dist(\overrightarrow{x_i}, C_k) = \frac{\sum_{d \in S_k} |x_{id} - \mu_{kd}|}{|S_k|}$$

Concernant la recherche des dimensions S_k associées aux médoïdes $\overrightarrow{\mu_k}$, celle-ci suit les étapes suivantes :

- calculer δ_k, le minimum des distances entre un médoïde donné et tous les autres :

$$\delta_k = min_{\{l \neq k\}} dist(\overrightarrow{\mu_l}, \overrightarrow{\mu_k})$$

- créer l'ensemble L_k représentant la *localité* de $\overrightarrow{\mu_k}$:

$$L_k = \{i \in D | dist(\overrightarrow{x_i}, \overrightarrow{\mu_k}) \leq \delta_k\}$$

- calculer $Dist_{kd}$ la distance moyenne des objets de L_k à $\overrightarrow{\mu_k}$ sur la dimension d :

$$Dist_{kd} = \frac{\sum_{i \in L_k} dist(x_{id}, \mu_{kd})}{|L_k|}$$

- calculer Y_k et σ_k les moyenne et déviation standard des valeurs $Dist_{kd}$:

$$Y_k = \frac{\sum_{d=1}^{M} Dist_{kd}}{M}$$

$$\sigma_k = \sqrt{\frac{\sum_{d=1}^{M} (Dist_{kd} - Y_k)^2}{M - 1}}$$

- calculer Z_{kd}, indicateur de la pertinence de la dimension d pour le cluster C_k :

$$Z_{kd} = \frac{Dist_{kd} - Y_k}{\sigma_k}$$

– puis trier les Z_{kd} par ordre croissant, sélectionner les deux minima pour chaque cluster C_k, puis les $K \times (L-2)$ minima restant, et associer les dimensions d correspondantes aux clusters C_k correspondants.

Enfin, une dernière phase de *raffinement* est menée. Tout d'abord, les dimensions associées aux clusters sont réévaluées, non plus uniquement en fonction des médoïdes, mais en fonction de tous les objets appartenant aux clusters. Puis les objets dont la distance à tous les médoïdes $\overrightarrow{\mu_k}$ est supérieure à la mesure Δ_k présentée ci-après sont considérés comme étant du bruit dans les données.

$$\Delta_k = min_{\{l \neq k\}} dist(\overrightarrow{\mu_l}, \overrightarrow{\mu_k})$$

B.3.2 FINDIT

Dans [Woo and Lee, 2002], les auteurs se basent également sur une recherche stochastique de K médoïdes représentant les clusters, mais pour cibler l'ensemble des dimensions qui leur correspond le mieux, ils utilisent une technique basée sur les plus proches voisins.

Une nouvelle mesure de distance, notée *dod* pour *Dimension-Oriented Distance*, est introduite. Celle-ci est basée sur l'hypothèse que dans un espace de grande dimensionalité, il est plus caractéristique que deux objets soient proches sur de nombreuses dimensions que très proches sur seulement quelques dimensions.

$$dod_\epsilon(\overrightarrow{x_i}, \overrightarrow{x_j}) = max\{dod_\epsilon(\overrightarrow{x_i} \rightarrow \overrightarrow{x_j}), dod_\epsilon(\overrightarrow{x_j} \rightarrow \overrightarrow{x_i})\}$$

$$dod_\epsilon(\overrightarrow{x_i} \rightarrow \overrightarrow{x_j}) = |S_i| - |\{d \in S_i \cap S_j \mid |x_{id} - x_{jd}| \leq \epsilon\}|$$

S_i dénote l'ensemble des dimensions associées à l'objet $\overrightarrow{x_i}$, et ϵ est un réel positif fixé proche de 0. Cette distance reflète ainsi le nombre de dimensions sur lesquelles deux objets sont distants de plus de ϵ.

Pour effectuer la sélection des dimensions associées aux objets, une nouvelle méthode appelée *Dimension Voting*, basée sur l'utilisation des plus proches voisins, est également introduite. Le principe est le suivant : pour un objet et une dimension donnés, si $\alpha\%$ de ses k plus proches voisins sont distants de moins de ϵ, alors la dimension est considérée comme caractéristique de l'objet.

Deux paramètres sont nécessaires à la méthode :

1. $C_{minsize}$, le nombre minimum d'objets que les clusters doivent contenir ;
2. $D_{mindist}$, la distance minimale séparant les clusters.

La méthode se divise ensuite en trois phases principales :

1. échantillonnage ;
2. formation des clusters ;
3. association des objets aux clusters.

Dans la première phase d'échantillonnage, deux ensembles sont créés : l'ensemble D des données et l'ensemble O des médoïdes. La formation des clusters suit ensuite les étapes suivantes :

1. déterminer l'ensemble des dimensions associées à chaque médoïde $\overrightarrow{\mu_k}$ de O par la méthode de *Dimension Voting* ;
2. assigner les objets $\overrightarrow{x_i}$ de D aux médoïdes dont ils sont les plus proches à la condition suivante :
$$dod_\epsilon(\overrightarrow{\mu_k} \rightarrow \overrightarrow{x_i}) = 0$$
les objets qui ne peuvent être assignés à aucun médoïde sont alors identifiés comme étant du bruit ;
3. regrouper les médoïdes $\overrightarrow{\mu_k}$ et $\overrightarrow{\mu_l}$ vérifiant la condition de proximité suivante :
$$dod_\epsilon(\overrightarrow{\mu_k} \rightarrow \overrightarrow{\mu_l}) < D_{mindist}$$
le médoïde résultant de la fusion est alors la moyenne des médoïdes initiaux pondérée par le nombre d'objets qui leur sont associés, et les dimensions conservées sont celles qui sont associées à plus de 95% des médoïdes concernés ;
4. regrouper les nouveaux médoïdes proches les uns des autres et supprimer ceux qui sont associés à moins d'un certain nombre d'objets (dépendant de l'échantillonnage effectué).

Cette méthode est alors exécutée 25 fois en faisant varier ϵ de 1 à 25% de l'intervalle de définition des objets, puis l'ensemble de médoïdes optimisant le critère suivant est conservé :
$$Q_\epsilon = \sum_{k=1}^{K} N_k \times |S_k|$$
Enfin, chaque objet de D est associé au cluster dont le médoïde est le plus proche.

B.3.3 DOC

Une autre possibilité présentée dans [Procopiuc et al., 2002] pour la formation des clusters projetés est de chercher à optimiser un critère qui se base sur une densité minimum à respecter par les clusters ainsi qu'une largeur maximum sur ses dimensions associées.

Étant donné $0 \leq \alpha \leq 1$ et $w \geq 0$, un cluster projeté α-dense de longueur w dans D est défini par un couple (D_k, S_k) où $D_k \subseteq D$ et $S_k \subseteq S$ tel que :

1. D_k est α-dense, c'est-à-dire que $N_k \geq \alpha \times N$;
2. $\forall\ d \in S_k, max_{\{i \in D_k\}} x_{id} - min_{\{i \in D_k\}} x_{id} \leq w$;
3. $\forall\ d \in S \backslash S_k, max_{\{i \in D_k\}} x_{id} - min_{\{i \in D_k\}} x_{id} > w$.

La qualité d'un cluster projeté est définie par $\mu(N_k, |S_k|)$, où $\mu : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ est une fonction monotone croissante sur chaque argument avec $\mu(0,0) = 0$: par exemple $\mu(a,b) = a(1/\beta)^b$.

La méthode proposée pour la formation des clusters est stochastique. D'abord un objet $\overrightarrow{\mu_k}$ est choisi aléatoirement comme graine et un ensemble d'objets D_k lui sont associés. L'ensemble des dimensions sur lequel le critère de largeur maximum est respecté est ensuite créé :

$$S_k = \{d \mid |x_{id} - \mu_{kd}| \leq w \ \forall \ i \in D_k\}$$

Puis D_k est mis à jour en sélectionnant les objets de D contenus dans l'hypercube de centre $\overrightarrow{\mu_k}$ et de largeur w sur S_k, et le cluster est conservé s'il est α-dense.

Le critère d'arrêt peut être défini de plusieurs façons : un certain pourcentage d'objets a été groupé, ou un certain nombre de clusters a été atteint, ou la mesure de qualité des clusters a trop diminuée. Les clusters optimisant la fonction μ, qui peuvent se chevaucher, seront finalement retournés.

B.3.4 HARP

Alors que les algorithmes existants laissent le soin à l'utilisateur de fixer certains paramètres du modèle (CLIQUE nécessite un seuil de densité, PROCLUS le nombre moyens de dimensions pertinentes des clusters, etc.), les auteurs de [Yip et al., 2003] proposent d'utiliser des seuils qui seront ajustés de façon dynamique afin d'éviter la spécification de paramètres par l'utilisateur.

Un indice de pertinence d'une dimension d pour un cluster C_k est d'abord définie comme le rapport entre la déviation standard locale et globale :

$$R(C_k, d) = 1 - \frac{\sigma^2(C_k, d)}{\sigma^2(D, d)}$$

Cet indice de pertinence est ensuite utilisé pour faire la sélection des dimensions pertinentes à considérer pour un cluster donné : celles dont la valeur est au dessus d'un certain seuil qui sera ajusté de façon dynamique. Cet indice est également utilisé dans le calcul de similarité entre clusters :

$$RSim(C_i, C_j) = \sum_{d \in S_k} R(C_k, d)$$

avec C_k le résultat de la fusion entre C_i et C_j, et S_k l'ensemble de ses dimensions associées.

Cependant, en utilisant cette mesure, deux cas de *mutual disagreement* peuvent arriver : lorsque sont effectuées des fusions entre deux clusters de tailles très différentes, ou entre deux clusters dont le nombre de dimensions associées est très différent. Une nouvelle mesure est donc introduite, qui empêchera de fusionner deux clusters si sa valeur est trop élevée :

$$MD(C_i, C_j) = \frac{1}{|S_i \cup S_j|} \sum_{d \in S_i \cup S_j} (1 - M(C_i, C_j, d))$$

$$M(C_i, C_j, d) = \begin{cases} 0 & \text{si } R(C_i, d|C_k) \leq 0 \text{ ou } R(C_j, d|C_k) \leq 0 \\ \frac{min(R(C_i,d|C_k), R(C_j,d|C_k))}{max(R(C_i,d|C_k), R(C_j,d|C_k))} & \text{sinon} \end{cases}$$

$$R(C_i, d|C_k) = 1 - \frac{(\mu(C_i, d) - \mu(C_k, d))^2 + \sigma^2(C_i, d)}{\sigma^2(D, d)}$$

L'algorithme utilisé est hiérarchique ascendant basé sur deux seuils dynamiques :
- S_{min} le nombre minimum de dimensions associées aux clusters ;
- et R_{min} la valeur minimum de l'indice de pertinence de ces dimensions.

Une dimension est alors associée à un cluster si son indice de pertinence est supérieur à R_{min}. Et deux clusters C_i et C_j sont autorisés à fusionner si le cluster résultant C_k possède au moins S_{min} dimensions associées (à noter que $S_k \subseteq S_i \cup S_j$).

Après avoir initialisé les deux seuils $S_{min} = M$ et $R_{min} = 1$, l'algorithme fonctionne en deux étapes itérées jusqu'à ce qu'un nombre de clusters cible soit atteint :

1. fusionner les deux clusters C_i et C_j ($\Rightarrow C_k$) maximisant leur similarité $RSim$, jusqu'à ce que plus aucune fusion ne soit possible :

$$|S_k| < S_{min} \text{ ou } \frac{\sum_{d \in S_k} R(C_k, a)}{|S_k|} \times MD(C_i, C_j) < R_{min} \; \forall \; (C_i, C_j, C_k)$$

2. diminuer les seuils :

$$S_{min} = S_{min} - 1 \text{ et } R_{min} = R_{min} - \frac{1}{M - 1}$$

B.3.5 LAC

Afin d'éviter la spécification de paramètres de la part de l'utilisateur, la méthode proposée dans [Domeniconi et al., 2004] et appelée LAC va associer à chaque dimension de chaque cluster un poids précisant son importance pour le cluster considéré, au lieu d'en sélectionner un sous-ensemble.

LAC se base pour cela sur l'algorithme K-means, et associe un vecteur de poids sur l'ensemble des dimensions de l'espace à chaque centroïde de cluster. Une notion de *cluster pondéré* est donc introduite : sous-ensemble des objets, associé à un vecteur de poids sur les dimensions de l'espace. Et l'algorithme procède comme suit :

1. démarrer avec K centroïdes initiaux : le premier est tiré aléatoirement, puis les suivants sélectionnés sont les plus éloignés possible de ceux déjà sélectionnés ;
2. initialiser les poids $w_{kd} = 1/\sqrt{M}$ de chaque cluster C_k sur chaque dimension de l'espace d ;
3. puis jusqu'à ce qu'il y ait convergence, c'est-à-dire que plus aucun objet ne change d'affectation de cluster d'une itération à l'autre :

– associer chaque objet au centroïde dont il est le plus proche, la notion de proximité dépendant de la matrice de poids W :

$$D_k = \{i \in D | k = ArgMin_l \ L_W(\overrightarrow{\mu_l}, \overrightarrow{x_i})\}$$

$$L_W(\overrightarrow{\mu_l}, \overrightarrow{x_i}) = \sqrt{\sum_{d=1}^{M} w_{ld} \times (\mu_{ld} - x_{id})^2}$$

– mettre à jour les poids des clusters de centroïde $\overrightarrow{\mu_k}$ sur chaque dimension d :

$$Y_{kd} = \frac{\sum_{i \in D_k} (\mu_{kd} - x_{id})^2}{N_k}$$

$$w_{kd} = \frac{exp(-h \times Y_{kd})}{\sqrt{\sum_{j=1}^{M} exp(-h \times 2 \times Y_{kj})}}$$

h permettant de donner plus ou moins d'importance à Y_{kd} dans la mise à jour des poids w_{kd}, et fixé à 9 expérimentalement ;

– mettre à jour les objets associés aux clusters :

$$D_k = \{i \in D | k = ArgMin_l \ L_W(\overrightarrow{\mu_l}, \overrightarrow{x_i})\}$$

– et mettre à jour les centroïdes des clusters :

$$\overrightarrow{\mu_k} = \frac{\sum_{i \in D_k} \overrightarrow{x_i}}{N_k}$$

B.3.6 CLTREE

Dans [Liu et al., 2000], une technique originale permettant d'adapter les arbres de décision au clustering a été proposée, qui permet ainsi d'associer aux clusters formés des sous-ensembles de dimensions qui leur sont propres, et de fournir en sortie un résultat compréhensible par l'utilisateur.

L'idée générale de la méthode appelée CLTREE est de considérer l'ensemble des objets de la base comme des exemples positifs, et de supposer l'espace uniformément distribué par des exemples négatifs. Le problème de partitionnement de l'espace des objets devient alors dans ce cadre un problème de classification supervisée, que les auteurs proposent de résoudre à l'aide des arbres de décision.

La procédure générale de la méthode consiste donc à contrôler l'ajout d'objets fictifs classés négativement pour une création aussi pertinente que possible de l'arbre de décision. Pour cela, les auteurs proposent de faire en sorte qu'il y ait toujours au moins autant que d'objets réels que d'objets fictifs dans la zone courante étudiée.

Initialement, autant d'objets fictifs répartis uniformément dans l'espace de description des objets sont donc ajoutés qu'il existe d'objets réels. Puis lorsque la méthode arrive dans une région de l'espace où il y a moins d'objets fictifs que d'objets réels, alors autant d'objets fictifs qu'il faut sont ajoutés pour atteindre le nombre d'objets réels.

Si au contraire la méthode arrive dans une région de l'espace dans laquelle il y a plus d'objets fictifs que d'objets réels, aucune action n'est menée, ce qui permet de cibler le bruit qui peut exister dans les données.

Les objets fictifs n'étant pas ajoutés physiquement, ils doivent être calculés *à la volée*. Ceci est fait simplement dans le calcul du *Gain d'Information* utilisé lors de la construction de l'arbre, cette mesure ne nécessitant que le nombre de membres de chaque classe dans chaque sous-partition créée par rapport à la dimension sélectionnée :

$$IG(E,d) = Entropie(E) - \sum_{s \in subset(d)} \frac{|E_s|}{|E|} Entropie(E_s)$$

avec E l'ensemble des objets concernés, d la dimension sélectionnée pour le partitionnement, s les sous-ensembles de d définissant la partition sur d, et $Entropie(E) = -\sum_{i=1}^{c} p_i \times log(p_i)$, où p_i est la proportion de membres de la classe i dans E, et c le nombre total de classes (égal à deux dans notre cas). Le nombre de membres de la classe négative associés au sous-ensemble s de d est donc calculé proportionnellement à la longueur relative de l'intervalle défini par s : $|s|/|d|$.

B.3.7 δ-cluster

Enfin, dans [Wang et al., 2002], les auteurs constatent que dans certains cas, comme par exemple dans des bases de données bio-informatiques ou en filtrage collaboratif, certains groupes ne sont pas caractérisés par des valeurs proches sur certaines dimensions mais sur une évolution identique de ces valeurs. L'objectif dans ce cadre est alors de regrouper les objets qui ont les mêmes *tendances* sur un sous-ensemble de dimensions.

Un nouveau modèle de δ-cluster est donc introduit pour définir la *cohérence* d'un sous-ensemble d'objets. Étant donnés $i, j \in D_k$ et $a, b \in S_k$, le $pScore$ de la matrice 2×2 correspondante est :

$$pScore\left(\begin{bmatrix} x_{ia} & x_{ib} \\ x_{ja} & x_{jb} \end{bmatrix}\right) = |(x_{ia} - x_{ib}) - (x_{ja} - x_{jb})|$$

Un couple (D_k, S_k) forme alors un δ-cluster si pour toute sous-matrice X de (D_k, S_k), $pScore(X) \leq \delta$ pour $\delta \geq 0$. Ainsi, toute sous-matrice (D'_k, S'_k) avec $D'_k \subseteq D_k$ et $S'_k \subseteq S_k$ est aussi un δ-cluster.

Trois paramètres sont nécessaires à la méthode : le seuil δ, le nombre minimum d'objets min_d et le nombre minimal de dimensions min_s considérés pour la formation des δ-clusters. La méthode est déterministe et descendante, et se donne comme objectif de caractériser les clusters par un nombre minimum de dimensions. Les ensembles de dimensions S_k associées aux clusters C_k doivent donc vérifier qu'aucun sous-ensemble $S'_k \supset S_k$ n'existe tel que (D_k, S'_k) soit également un δ-cluster.

Étant donné un ensemble d'attributs S_k, deux objets $\overrightarrow{x_i}$ et $\overrightarrow{x_j}$ forment un δ-cluster sur S_k si leurs valeurs sur l'ensemble des dimensions de S_k ont une distance inférieure à δ :

$$|x_{id} - x_{jd}| < \delta \ \forall \ d \in S_k$$

Pour cibler l'ensemble maximal S_k des dimensions associées à deux objets $\overrightarrow{x_i}$ et $\overrightarrow{x_j}$, on forme dans un premier temps la séquence correspondant à l'ensemble des différences de leurs valeurs sur S rangées dans l'ordre croissant :

$$\overrightarrow{S}(\overrightarrow{x_i}, \overrightarrow{x_j}, S) = s_1 ... s_M$$

L'ensemble maximal S_k des dimensions correspond alors à l'ensemble des dimensions présentes dans la sous-séquence de taille maximale de cette séquence :

$$\overrightarrow{S}(\overrightarrow{x_i}, \overrightarrow{x_j}, S_k) = \{s_i ... s_j | j > i \text{ et } s_j - s_i \leq \delta \text{ et } s_{j+1} - s_i > \delta \text{ et } s_j - s_{i-1} > \delta\}$$

Une fenêtre, définie par une borne minimale et une borne maximale, est donc déplacée sur la séquence $\overrightarrow{S}(\overrightarrow{x_i}, \overrightarrow{x_j}, S)$. Si les valeurs aux extrémités de la fenêtre ont une distance inférieure à δ, alors la borne maximale est incrémentée. Sinon, la sous-séquence précédente est stockée si sa taille était supérieure à min_s, et c'est la borne minimale qui est incrémentée.

Ces ensembles servent alors de base pour la formation finale des δ-clusters, les paramètres min_d et min_s servant à faire le tri parmi les différents candidats. Outre la difficile spécification de ces paramètres par l'utilisateur, la problématique la plus importante de cette méthode est évidemment sa complexité, en $O(N^2 \times M \log M + M^2 \times N \log N)$ en fonction du nombre d'objets N et du nombre de dimensions M. Par contre, c'est la seule à avoir pris en compte la problématique des clusters n'étant pas définis par des valeurs identiques sur certaines dimensions, mais par des évolutions identiques de ces valeurs.

B.4 Bilan

L'objectif du subspace clustering est en fait de trouver un bon compromis entre :

1. minimiser les distances entre les valeurs des membres d'un cluster donné sur ses dimensions associées,
2. maximiser le nombre de dimensions associées aux clusters,
3. et maximiser le nombre de membres des clusters.

Intuitivement, de faibles distances entre les valeurs des membres d'un même cluster sur ses dimensions associées indiquent que les membres de ce cluster s'accordent sur de petits intervalles de valeurs. Un nombre important de dimensions associées indique que les membres du cluster sont similaires dans un sous-espace de grande dimension, et donc qu'ils ont une probabilité forte d'appartenir au même concept. Et un nombre de membres important dans un cluster indique un support important sur les dimensions sélectionnées, et c'est donc peu probable que les faibles distances entre ces objets soient le fruit du hasard. Mais la difficulté du problème vient du fait que ces contraintes vont dans des sens opposés :

- pour un nombre fixe d'objets, minimiser les distances entre valeurs sur les dimensions sélectionnées va tendre à en sélectionner le moins possible ;

- pour un nombre fixe de dimensions, minimiser les distances entre valeurs des objets sélectionnés va tendre à en sélectionner le moins possible ;
- et pour une valeur maximum fixée de distance entre membres d'un même cluster sur ses dimensions associées, maximiser le nombre de dimensions associées va tendre à ne considérer que peu d'objets, et inversement, maximiser le nombre d'objets considérés va tendre à minimiser le nombre de dimensions associées.

Face à ce difficile problème d'optimisation, les différentes méthodes de subspace clustering que nous avons présentées se basent sur des techniques différentes : les méthodes de subspace clustering ascendantes sur les dimensions, telle CLIQUE, laissent le soin à l'utilisateur de fixer le premier critère de la densité minimum à respecter par les différents clusters, et cherchent alors les clusters qui maximisent le nombre de dimensions associées aux clusters, correspondant au second critère ; PROCLUS, lui, demande à l'utilisateur de fixer le nombre de dimensions associées aux clusters, puis cherche les dimensions qui minimisent le premier critère ; quant à DOC, il demande à l'utilisateur de fixer le premier critère de densité minimum à respecter pour la formation des clusters, puis cherche les clusters qui maximisent le nombre de dimensions associées aux clusters et le nombre de membres de ces clusters (critères 2 et 3) ; HARP va lui utiliser des seuils dynamiques afin d'éviter de requérir de la connaissance a priori de la part de l'utilisateur ; enfin LAC va lui faire le choix de pondérer l'importance de chaque dimension afin d'éviter d'avoir à en sélectionner un sous-ensemble.

Annexe C

Algorithme Tuareg

C.1 Méthode de partitionnement de Fisher

Soit $E = \{v_1, ..., v_n\}$ un ensemble ordonné, de taille n, de valeurs à partitionner en k sous-ensembles. L'algorithme de partitionnement proposé dans [Diday et al., 1982a] permet de trouver la partition optimale de cet ensemble, pour un nombre k de clusters fixé, vis-à-vis de la somme (W) des inerties intra-clusters (I) des clusters créés.

On identifie un cluster c_i par un couple d'identifiants minimum m_i et maximum M_i sur E. L'inertie d'un cluster $c_i = \{v_{m_i}, ..., v_{M_i}\}$ est alors donnée par :

$$I(c_i) = \sum_{m=m_i}^{M_i} (v_m - G_i)^2$$

où G_i est le centre de gravité du cluster :

$$G_i = \frac{\sum_{m=m_i}^{M_i} v_m}{M_i - m_i}$$

On note $W(P_l)$ la somme des inerties des clusters c_i, pour $i \in [1, l]$, contenus dans la partition P_l de taille l :

$$W(P_l) = \sum_{i=1}^{l} I(c_i)$$

Inspiré de la programmation dynamique, l'algorithme de Fisher consiste à calculer par récurrence une suite de partitions optimales P_l^i de $\{v_i, ..., v_n\}$ en l classes. Le déroulement de l'algorithme peut être décrit de la façon suivante :

- on pose $P_1^i = \{v_i, ..., v_n\}$ pour $i \in [1, n]$;
- pour $l \in [2, k-1]$, on calcule les partitions

$$P_l^i = (\{v_i, ..., v_j\}, P_{l-1}^{j+1}) \text{ pour } i \in [1, n-l+1]$$

où la partition P_{l-1}^{j+1} à $l-1$ classes de $\{v_{j+1}, ..., v_n\}$ a été calculée à l'étape précédente $l-1$ et où j est choisi dans $[i, n-l+1]$ de façon à ce que $I(\{v_i, ..., v_j\}) + W(P_{l-1}^{j+1})$ soit minimum ;

– enfin, à l'étape k, on construit la partition

$$P_k^1 = (\{v_1, ..., v_i\}, P_{k-1}^{i+1})$$

où i est choisi dans $[1, n-k+1]$ de façon à minimiser $I(\{v_1, ..., v_i\}) + W(P_{k-1}^{i+1})$.

La figure C.1 montre un exemple d'exécution de cet algorithme, pour l'ensemble $E = [0, 1, 8, 9]$, et $k = 3$.

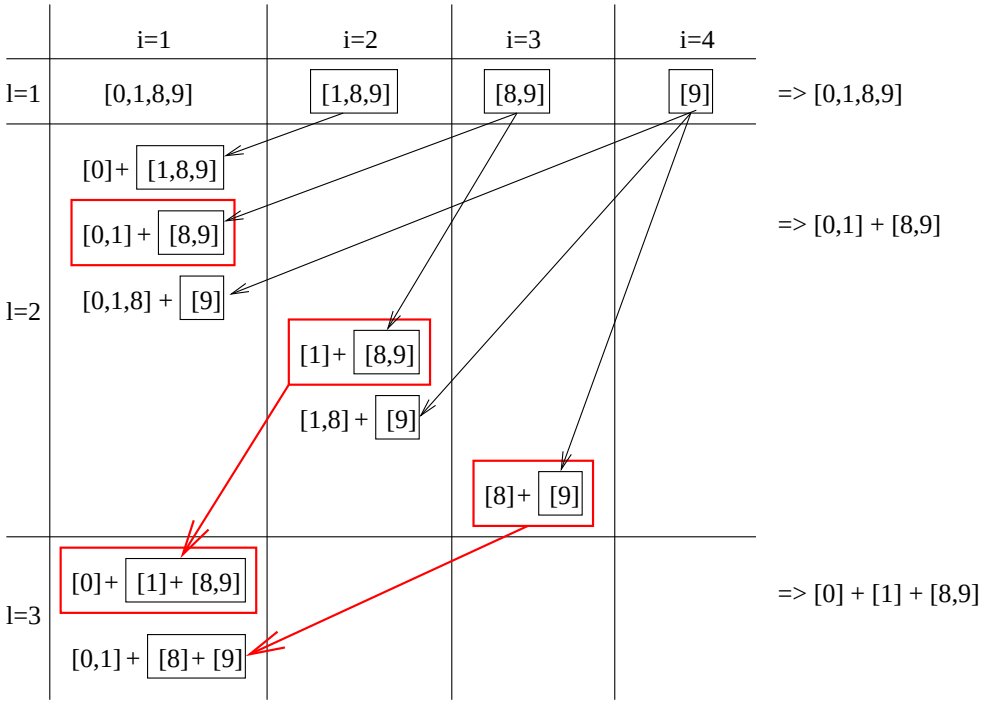


FIG. C.1 – Partitionnement de Fisher de l'ensemble [0,1,8,9] pour K=3.

Annexe D

Algorithme SuSE

D.1 Démonstrations

Dans cette section, nous justifions de manière théorique les équations que nous utilisons dans les algorithmes **SSC** et **SuSE** présentés dans le chapitre 5.

Ces deux méthodes statistiques sont basées sur l'hypothèse que les données ont été générées selon un mélange de distributions de probabilités, la problématique étant alors de trouver, en fonction des données observées, les paramètres de ces distributions, ainsi que les probabilités du mélange, et les paramètres cachés du modèle correspondant aux affectations des objets aux différentes composantes du mélange.

Le modèle que nous avons construit est lui basé sur l'hypothèse que les données ont été générées selon des distributions indépendantes sur chaque dimension, gaussienne sur dimension numérique et multinomiale sur dimension catégorielle.

Le modèle θ que nous utilisons pour représenter les données consiste donc en un mélange de K distributions θ_k pour $k \in [1, K]$, caractérisées par leur centre μ_{kd} et leur variance σ_{kd} sur toute dimension numérique d, par un vecteur de fréquences $\overrightarrow{Freqs_{kd}}$ sur l'ensemble des modalités possibles de toute dimension catégorielle d, et par leur probabilité d'apparition π_k.

$$\theta = (\theta_1, ..., \theta_K)$$

$$\theta_k = (\overrightarrow{\mu_k}, \overrightarrow{\sigma_k}, Freqs_k, \pi_k)$$

$$\sum_{k=1}^{K} \pi_k = 1$$

Les appartenances des objets aux clusters sont alors vues comme des paramètres cachés du modèle. On pose $z_{ik} = 1$ si l'objet $\overrightarrow{x_i}$ a été généré par la distribution θ_k, et $z_{ik} = 0$ sinon. L'ensemble $D_z = \{\overrightarrow{z_i} | 1 \leq i \leq N\}$ forme alors l'ensemble des données cachées, $D_x = \{\overrightarrow{x_i} | 1 \leq i \leq N\}$ l'ensemble des données observées, et $D_y = \{y_i | 1 \leq i \leq N\}$ l'ensemble complet des données, avec $y_i = (\overrightarrow{x_i}, \overrightarrow{z_i})$.

L'objectif est alors de maximiser la vraisemblance du modèle θ par rapport aux données complètes D_y, notée $L(\theta|D_y)$. On parle d'*estimation par maximum de vraisemblance*. Si l'on fait l'hypothèse que les objets ont été générés indépendamment les uns des autres, on a :

$$L(\theta|D_y) = P(D_y|\theta) = \prod_{i=1}^{N} P(y_i|\theta)$$

L'utilisation du logarithme de cette fonction permet de considérer une somme de termes au lieu d'un produit, et maximiser $L(\theta|D_y)$ équivaut à maximiser son logarithme.

$$\log L(\theta|D_y) = \sum_{i=1}^{N} \log P(y_i|\theta)$$

La méthode que nous utilisons pour résoudre ce problème d'optimisation est l'algorithme *EM*, qui consiste en une succession de deux phases permettant d'optimiser le modèle θ à chaque étape :

1. étape E (Estimation) : évaluer les paramètres cachés du modèle D_z en fonction des paramètres courants du modèle θ ;
2. étape M (Maximisation) : mettre à jour les paramètres du modèle θ en fonction des nouveaux paramètres cachés D_z.

Plus formellement, la première étape d'Estimation de l'algorithme EM revient à supposer le modèle θ connu et à chercher l'ensemble des données cachées D_z optimal sous ce modèle, ce qui revient à estimer :

$$Q(\theta) = E_z\{\log P(D_y|\theta)|D_x, \theta\}$$

Étant donnée l'hypothèse d'indépendance des objets, on a :

$$P(D_y|\theta) = \prod_{i=1}^{N} P(y_i|\theta)$$

$$\begin{aligned} P(y_i|\theta) &= P(\overrightarrow{x_i}, \overrightarrow{z_i}|\theta) \\ &= \pi_k \times P(\overrightarrow{x_i}|\theta_k) \text{ pour } k \text{ tel que } z_{ik} = 1 \\ &= \textstyle\prod_{k=1}^{K} (\pi_k \times P(\overrightarrow{x_i}|\theta_k))^{z_{ik}} \end{aligned}$$

$$P(D_y|\theta) = \prod_{i=1}^{N} \prod_{k=1}^{K} (\pi_k \times P(\overrightarrow{x_i}|\theta_k))^{z_{ik}}$$

$$\begin{aligned} \log P(D_y|\theta) &= \textstyle\sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \times \log(\pi_k \times P(\overrightarrow{x_i}|\theta_k)) \\ &= \textstyle\sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \times \log P(\overrightarrow{x_i}|\theta_k) + \sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \times \log \pi_k \end{aligned}$$

$$\begin{aligned} Q(\theta) &= E_z\{\log P(D_y|\theta)|D_x, \theta\} \\ &= \textstyle\sum_{i=1}^{N} \sum_{k=1}^{K} E_z\{z_{ik}|D_x, \theta\} \times \log P(\overrightarrow{x_i}|\theta_k) + \sum_{i=1}^{N} \sum_{k=1}^{K} E_z\{z_{ik}|D_x, \theta\} \times \log \pi_k \end{aligned}$$

$$\begin{aligned} \overline{z}_{ik} &= E_z\{z_{ik}|D_x,\theta\} \\ &= 0 \times P(z_{ik}=0|D_x,\theta) + 1 \times P(z_{ik}=1|D_x,\theta) \\ &= P(z_{ik}=1|\overrightarrow{x_i},\theta) \\ &= \frac{\pi_k \times P(\overrightarrow{x_i}|\theta_k)}{\sum_j \pi_j \times P(\overrightarrow{x_i}|\theta_j)} \end{aligned} \tag{D.1}$$

Finalement, la phase d'Estimation de l'algorithme EM consiste donc à estimer ces probabilités $\overline{z}_{ik}$ étant donné le modèle courant θ supposé connu, selon l'équation D.1.

Dans la seconde phase de Maximisation de l'algorithme EM, l'hypothèse inverse est considérée : on suppose connues les probabilités d'appartenance des objets aux clusters $\overline{z}_{ik}$, et on cherche à maximiser la fonction Q sous la contrainte que $\sum_{k=1}^{K} \pi_k = 1$. Pour cela, on construit donc la nouvelle fonction Q' suivante, utilisant le multiplicateur de Lagrange λ :

$$Q'(\theta) = \sum_{i=1}^{N}\sum_{k=1}^{K} \overline{z}_{ik} \times \log P(\overrightarrow{x_i}|\theta_k) + \sum_{i=1}^{N}\sum_{k=1}^{K} \overline{z}_{ik} \times \log(\pi_k) + \lambda \times (1 - \sum_{k=1}^{K} \pi_k)$$

L'optimum de cette fonction est atteint lorsque sa dérivée par rapport aux paramètres inconnus du modèle est nulle. Voyons d'abord ce que cela entraîne dans la mise à jour des paramètres π_k.

$$\frac{\partial Q'}{\partial \pi_k} = \sum_{i=1}^{N} \frac{\overline{z}_{ik}}{\pi_k} - \lambda = 0$$

$$\pi_k = \frac{\sum_{i=1}^{N} \overline{z}_{ik}}{\lambda}$$

$$\sum_{k=1}^{K} \pi_k = 1 = \frac{\sum_{k=1}^{K}\sum_{i=1}^{N} \overline{z}_{ik}}{\lambda}$$

$$\lambda = \sum_{k=1}^{K}\sum_{i=1}^{N} \overline{z}_{ik} = N$$

$$\text{Donc } \pi_k = \frac{\sum_{i=1}^{N} \overline{z}_{ik}}{N} \tag{D.2}$$

Voyons ensuite la mise à jour des paramètres dans le cas que nous considérons où la distribution est supposée gaussienne sur les dimensions numériques et multinomiale sur les dimensions catégorielles.

$$P(\overrightarrow{x_i}|\theta_k) = \prod_{d \in S_k} P(x_{id}|\theta_{kd})$$

$$P(x_{id}|\theta_{kd}) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma_{kd}} e^{-\frac{1}{2}\left(\frac{x_{id}-\mu_{kd}}{\sigma_{kd}}\right)^2} & \text{si } d \text{ numérique} \\ Freqs_{kd}(x_{id}) & \text{si } d \text{ catégorielle} \end{cases}$$

$$\log P(\overrightarrow{x_i}|\theta_k) = \sum_{d \in S_k} \log P(x_{id}|\theta_{kd})$$

$$\log P(x_{id}|\theta_{kd}) = \begin{cases} \log \frac{1}{\sqrt{2\pi}\sigma_{kd}} - \frac{1}{2}\left(\frac{x_{id}-\mu_{kd}}{\sigma_{kd}}\right)^2 & \text{si } d \text{ numérique} \\ \log Freqs_{kd}(x_{id}) & \text{si } d \text{ catégorielle} \end{cases}$$

Commençons par calculer la dérivée de la fonction Q' par rapport à μ_{kd}.

$$\frac{\partial}{\partial \mu_{kd}} \log P(\overrightarrow{x_i}|\theta_k) = \frac{1}{\sigma_{kd}^2}(x_{id} - \mu_{kd})$$

$$\frac{\partial Q'}{\partial \mu_{kd}} = \sum_{i=1}^{N} \overline{z}_{ik} \frac{1}{\sigma_{kd}^2}(x_{id} - \mu_{kd}) = \frac{1}{\sigma_{kd}^2} \sum_{i=1}^{N} \overline{z}_{ik} \times (x_{id} - \mu_{kd}) = 0$$

$$\text{Donc } \sum_{i=1}^{N} \overline{z}_{ik} \times x_{id} = \sum_{i=1}^{N} \overline{z}_{ik} \times \mu_{kd}$$

$$\text{Donc } \mu_{kd} = \frac{\sum_{i=1}^{N} \overline{z}_{ik} \times x_{id}}{\sum_{i=1}^{N} \overline{z}_{ik}} \tag{D.3}$$

Calculons ensuite la dérivée de Q' par rapport à σ_{kd}.

$$\frac{\partial}{\partial \sigma_{kd}} \log \frac{1}{\sqrt{2\pi}\sigma_{kd}} = -\frac{1}{\sigma_{kd}}$$

$$\frac{\partial}{\partial \sigma_{kd}} \left\{ -\frac{1}{2}\left(\frac{x_{id} - \mu_{kd}}{\sigma_{kd}}\right)^2 \right\} = \frac{(x_{id} - \mu_{kd})^2}{\sigma_{kd}^3}$$

$$\text{Donc } \frac{\partial Q'}{\partial \sigma_{kd}} = \sum_{i=1}^{N} \overline{z}_{ik} \left(-\frac{1}{\sigma_{kd}} + \frac{(x_{id} - \mu_{kd})^2}{\sigma_{kd}^3} \right) = 0$$

$$\sum_{i=1}^{N} \overline{z}_{ik} \left(\frac{(x_{id} - \mu_{kd})^2}{\sigma_{kd}^2} - 1 \right) = 0$$

$$\sum_{i=1}^{N} \overline{z}_{ik} \frac{(x_{id} - \mu_{kd})^2}{\sigma_{kd}^2} = \sum_{i=1}^{N} \overline{z}_{ik}$$

$$\text{Et finalement } \sigma_{kd}^2 = \frac{\sum_{i=1}^{N} \overline{z}_{ik} \times (x_{id} - \mu_{kd})^2}{\sum_{i=1}^{N} \overline{z}_{ik}} \tag{D.4}$$

Enfin, calculons la dérivée de Q' par rapport à $Freqs_{kd}(mod)$.

$$\frac{\partial}{\partial Freqs_{kd}(mod)} \log P(\overrightarrow{x_i}|\theta_k) = I_{\{x_{id}=mod\}} = \begin{cases} \frac{1}{Freqs_{kd}(mod)} & \text{si } x_{id} = mod \\ 0 & \text{sinon} \end{cases}$$

À cela on ajoute la contrainte que pour tout cluster C_k avec $k \in [1, K]$, et toute dimension catégorielle d, on a :

$$\sum_{mod \in Modalites_d} Freqs_{kd}(mod) = 1$$

Il faut donc ajouter le terme suivant C' à Q' :

$$C' = \sum_{k=1}^{K} \sum_{d \text{ catégorielle}} \sum_{mod \in Modalites_d} \lambda_{kd} \times (1 - Freqs_{kd}(mod))$$

$$\frac{\partial C'}{\partial Freqs_{kd}(mod)} = -\lambda_{kd}$$

$$\frac{\partial Q'}{\partial Freqs_{kd}(mod)} = \sum_{i=1}^{N} \overline{z}_{ik} \times I_{\{x_{id}=mod\}} - \lambda_{kd} = \sum_{\{1 \leq i \leq N | x_{id}=mod\}} \frac{\overline{z}_{ik}}{Freqs_{kd}(mod)} - \lambda_{kd} = 0$$

$$\text{Donc } Freqs_{kd}(mod) = \frac{\sum_{\{1 \leq i \leq N | x_{id}=mod\}} \overline{z}_{ik}}{\lambda_{kd}}$$

$$\text{Or } \sum_{mod \in Modalites_d} Freqs_{kd}(mod) = 1$$

$$\sum_{mod \in Modalites_d} \frac{\sum_{\{1 \leq i \leq N | x_{id}=mod\}} \overline{z}_{ik}}{\lambda_{kd}} = \frac{\sum_{i=1}^{N} \overline{z}_{ik}}{\lambda_{kd}} = 1$$

$$\text{Donc } \lambda_{kd} = \sum_{i=1}^{N} \overline{z}_{ik}$$

$$\text{Et donc } Freqs_{kd}(mod) = \frac{\sum_{\{1 \leq i \leq N | x_{id}=mod\}} \overline{z}_{ik}}{\sum_{i=1}^{N} \overline{z}_{ik}} \tag{D.5}$$

Finalement, la phase de Maximisation de l'algorithme EM revient à mettre à jour les paramètres du modèle selon les équations D.2, D.3, D.4 et D.5.

D.2 Détails d'implémentation

Dans des problèmes contenant de nombreuses dimensions, les calculs de probabilité peuvent rapidement dépasser les capacités machines. Pour ces calculs, il faudra donc utiliser des nombres sous forme scientifique (mantisse et exposant).

Pour éviter qu'une probabilité nulle sur une dimension catégorielle n'annule une probabilité globale, on fixe la fréquence minimale d'une modalité à 10^{-300}.

De même, et pour éviter les divisions par 0 lorsque la déviation standard des membres d'un cluster sur une dimension numérique est égale à 0, on fixe la déviation standard minimale à 0.0269, ce qui correspond à $P(x_{id}|\theta_{kd}) = 10^{-300}$ lorsque $x_{id} - \mu_{kd} = 1$.

Les valeurs manquantes sont ignorées dans les calculs. Ainsi, l'initialisation de μ_{kd} avec la possibilité de valeurs manquantes se réécrit comme suit :

$$\mu_{kd} = \frac{\sum_{\{i \in D_k | x_{id} \text{ défini}\}} x_{id}}{|\{i \in D_k | x_{id} \text{ défini}\}|}$$

D.3 Utilisation dans le logiciel Pertinence

L'utilisation de l'algorithme **SuSE** dans Pertinence Rule Maker ^{TM} permet à l'utilisateur d'appréhender un nouveau jeu de données. Il lui permet de découvrir la topologie de sa base, d'observer la répartition des objets dans leur espace de description, en lui mettant en avant les dimensions sur lesquels les groupes d'objets se distinguent le plus.

Il peut par exemple lui permettre de découvrir des points isolés, des corrélations entre dimensions, des groupes d'objets bien séparés, des zones de l'espace très denses, ou bien des zones de densité différentes. Il peut donc être utilisé pour des problèmes typiquement non supervisés. Mais il peut également mettre en avant des liens entre les classes et les zones de l'espace, s'il existe un lien entre les clusters et les classes. L'utilisateur peut être intéressé par étudier la topologie générale de la base ou bien la topologie par classe.

À partir d'une telle étude, il peut vouloir sauvegarder directement les règles apprises identifiées comme porteuses de sens. Il peut aussi vouloir séparer le jeu de données initial en sous-jeux de données selon les clusters formés. Il peut vouloir créer de nouveaux attributs en associant par exemple à tout objet de la base initiale un identifiant du cluster auquel il est associé. Il peut vouloir supprimer du jeu de données quelques objets isolés. Ou il peut vouloir concentrer sa recherche sur les dimensions pointées comme discriminantes au niveau de la topologie.

Enfin, nous proposons ici plusieurs indicateurs qui peuvent être associés aux résultats du clustering, permettant ainsi à l'utilisateur d'appréhender au mieux les résultats produits :

1. La vraisemblance du modèle par rapport aux données est fournie par la mesure $LL(\theta|D)$. Mais cette mesure prend des valeurs qui ne sont pas définies dans un intervalle spécifique. Afin d'obtenir une valeur qui ait du sens pour l'utilisateur, on peut alors utiliser le rapport entre la valeur du critère BIC pour la partition

courante, et celle calculée pour une partition ne contenant qu'un seul cluster. La valeur correspondra alors à un taux d'intérêt du partitionnement.

2. La plupart du temps, la méthode sera utilisée en pré-traitement d'une recherche supervisée. Pour mesurer l'adéquation entre les clusters et les classes, un critère de correspondance entre les classes et les clusters, telle l'entropie, peut donc être calculé et fourni à l'utilisateur.
3. La méthode étant exécutée plusieurs fois avec des initialisations aléatoires, on peut en profiter pour évaluer la stabilité du clustering. L'idée de base est la suivante : si les partitions changent du tout au tout à chaque exécution, le jeu de données n'a probablement pas de partition naturelle et celle qui est obtenue est contestable ; si au contraire certains éléments sont groupés dans toutes les partitions, alors on peut penser que ces éléments sont naturellement groupés. On peut donc utiliser une matrice de compatibilité entre objets, spécifiant s'ils se retrouvent souvent groupés ensemble lors des différentes exécutions. À chaque exécution, la compatibilité entre deux objets appartenant au même cluster est incrémentée. Et à la fin des différentes exécutions, on calcule pour chaque cluster de la partition finale la moyenne des compatibilités entre ses membres, divisée par le nombre d'itérations pour obtenir un taux.
4. Le poids moyen des dimensions pour chaque cluster fournit également de l'information sur les clusters produits : plus ce poids est élevé, plus ce cluster est spécifique, et plus il est probable qu'il ne soit pas dû au hasard.
5. Enfin, le nombre d'objets entrés dans les règles lors du passage du modèle aux règles fournit également de l'information : il donne une idée du taux de *caricature* des règles par rapport au modèle appris, et de la proximité des clusters entre eux.

Bibliographie

[Aggarwal et al., 1999] Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., and Park, J. S. (1999). Fast algorithms for projected clustering. In *ACM SIGMOD International Conference on Management of Data*, pages 61–72.

[Agrawal et al., 1998] Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *ACM SIGMOD International Conference on Management of Data*, pages 94–105.

[Akaike, 1970] Akaike, H. (1970). Statistical predictor identification. *Annals of the Institute for Statistical Mathematics*, 22 :203–217.

[Ali and Pazzani, 1996] Ali, K. M. and Pazzani, M. J. (1996). Error reduction through learning multiple descriptions. *Machine Learning*, 24(3) :173–202.

[Alpaydin, 1999] Alpaydin, E. (1999). Combined 5x2cv F-test for comparing supervised classification learning algorithms. *Neural Computation*, 11(8) :1885–1892.

[Ambroise and Govaert, 2000] Ambroise, C. and Govaert, G. (2000). EM algorithm for partially known labels. *7th Conference of the International Federation of Classication Societies*, pages 161–166.

[Ankerst et al., 1999] Ankerst, M., Breunig, M., Kriegel, H.-P., and Sander, J. (1999). OPTICS : Ordering points to identify the clustering structure. In *ACM SIGMOD International Conference on Management of Data*, pages 49–60.

[Apte et al., 2002] Apte, C. V., Natarajan, R., Pednault, E. P. D., and Tipu, F. A. (2002). A probabilistic estimaton framework for predictive model analytics. *IBM Systems Journal*, 41(3).

[Bengio and Grandvalet, 2004] Bengio, Y. and Grandvalet, Y. (2004). No unbiased estimator of the variance of k-fold cross-validation. *Journal of Machine Learning Research*, 5 :1089–1105.

[Berkhin, 2002] Berkhin, P. (2002). Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, California.

[Biernacki et al., 2000] Biernacki, C., Celeux, G., and Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7) :719–725.

[Biernacki and Govaert, 1998] Biernacki, C. and Govaert, G. (1998). Choosing models in model-based clustering and discriminant analysis. Technical Report 3509, Unité de recherche INRIA Rhône-Alpes, Montbonnot St Martin, France.

[Blake and Merz, 1998] Blake, C. and Merz, C. (1998). UCI repository of machine learning databases [http ://www.ics.uci.edu/∼mlearn/MLRepository.html].

[Bock and Diday, 2000] Bock, H. H. and Diday, E. (2000). *Exploratory Methods for Extracting Statistical Information from Complex Data*. Springer, Heidelberg.

[Bradley et al., 1998] Bradley, P., Fayyad, U., and Reina, C. (1998). Scaling EM (Expectation-Maximization) clustering to large databases. Technical report, Microsoft Research.

[Breiman, 1996a] Breiman, L. (1996a). Bagging predictors. *Machine Learning*, 24(2) :123–140.

[Breiman, 1996b] Breiman, L. (1996b). Bias, variance, and arcing classifiers. Technical Report 460, Statistics Department, University of California.

[Brézellec and Didier, 2001] Brézellec, P. and Didier, G. (2001). GIZMO : un algorithme de grille cherchant des clusters homogènes. *3ème Conférence francophone sur l'Apprentissage automatique*, pages 101–116.

[Candillier et al., 2004] Candillier, L., Tellier, I., and Torre, F. (2004). Tuareg : Classification non supervisée contextualisée. In Liquière, M. and Sebban, M., editors, *6ème Conférence francophone sur l'Apprentissage automatique*, pages 159–174.

[Candillier et al., 2005a] Candillier, L., Tellier, I., and Torre, F. (2005a). Transforming XML trees for efficient classification and clustering. INEX 2005 Workshop on Mining XML documents.

[Candillier et al., 2005b] Candillier, L., Tellier, I., Torre, F., and Bousquet, O. (2005b). SSC : Statistical Subspace Clustering. In Perner, P. and Imiya, A., editors, *4th International Conference on Machine Learning and Data Mining in Pattern Recognition*, volume LNAI 3587 of *Lecture Notes in Computer Science*, pages 100–109.

[Candillier et al., 2006a] Candillier, L., Tellier, I., Torre, F., and Bousquet, O. (2006a). Cascade evaluation of clustering algorithms. In Fürnkranz, J., Scheffer, T., and Spiliopoulou, M., editors, *17th European Conference on Machine Learning*, volume LNAI 4212 of *Lecture Notes in Computer Science*, pages 574–581.

[Candillier et al., 2006b] Candillier, L., Tellier, I., Torre, F., and Bousquet, O. (2006b). SuSE : Subspace Selection embedded in an EM algorithm. In Miclet, L., editor, *8ème Conférence francophone sur l'Apprentissage automatique*, pages 331–345.

[Celeux and Govaert, 1992] Celeux, G. and Govaert, G. (1992). A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14 :315–332.

[Cheng et al., 1999] Cheng, C. H., Fu, A. W.-C., and Zhang, Y. (1999). Entropy-based subspace clustering for mining numerical data. In *Knowledge Discovery and Data Mining*, pages 84–93.

[Costa et al., 2004] Costa, G., Manco, G., Ortale, R., and Tagarelli, A. (2004). A tree-based approach to clustering XML documents by structure. Technical report, Institute of Italian National Research Council, Rende, Italy.

[Dalamagas et al., 2004] Dalamagas, T., Cheng, T., Winkel, K.-J., and Sellis, T. (2004). Clustering XML documents by structure. In *3rd Hellenic Conference on Artificial Intelligence*.

[Dash et al., 2002] Dash, M., Choi, K., Scheuermann, P., and Liu, H. (2002). Feature selection for clustering - a filter solution. *IEEE International Conference on Data Mining*, pages 115–122.

[Dempster et al., 1977] Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1) :1–38.

[Denoyer et al., 2006] Denoyer, L., Gallinari, P., and Vercoustre, A. M. (2006). XML Mining Challenge at INEX 2005. Technical report, University of Paris VI, INRIA.

[Diday et al., 1980] Diday, E., Govaert, G., Lechevallier, Y., and Sidi, J. (1980). Clustering in pattern recognition. In *5th International Conference on Pattern Recognition*.

[Diday et al., 1982a] Diday, E., Lemaire, J., Pouget, J., and Testu, F. (1982a). *Un algorithme de partitionnement optimal dans le cas d'une variable unique*, pages 129–132. Dunod.

[Diday et al., 1982b] Diday, E., Lemaire, J., Pouget, J., and Testu, F. (1982b). *Un algorithme de type nuées dynamiques*, pages 117–123. Dunod.

[Diday and Vrac, 2005] Diday, E. and Vrac, M. (2005). Mixture decomposition of distributions by copulas in the symbolic data analysis framework. *Discrete Applied Mathematics*, 147 :27–41.

[Dietterich, 1998] Dietterich, T. G. (1998). Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7) :1895–1923.

[Dietterich, 2000] Dietterich, T. G. (2000). Ensemble methods in machine learning. In Kittler, J. and Roli, F., editors, *1st International Workshop on Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15.

[Domeniconi et al., 2004] Domeniconi, C., Papadopoulos, D., Gunopolos, D., and Ma, S. (2004). Subspace clustering of high dimensional data. In *SIAM International Conference on Data Mining*.

[Doucet and Ahonen-Myka, 2002] Doucet, A. and Ahonen-Myka, H. (2002). Naïve clustering of a large XML document collection. In *1st Annual Workshop of the Initiative for the Evaluation of XML retrieval*.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

[Fisher, 1987] Fisher, D. (1987). Knowledge acquisition via incremental conceptual clustering. In *Machine Learning*, pages 139–172.

[Flesca et al., 2002] Flesca, S., Manco, G., Masciari, E., Pontieri, L., and Pugliese, A. (2002). Detecting structural similarities between XML documents. In *5th International Workshop on The Web and Databases*.

[Freund and Schapire, 1996] Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156.

[Friedman and Meulman, 2004] Friedman, J. H. and Meulman, J. J. (2004). Clustering objects on subsets of attributes. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 66(4) :1–25.

[Gama and Brazdil, 2000] Gama, J. and Brazdil, P. (2000). Cascade generalization. *Machine Learning*, 41(3) :315–343.

[Gennari et al., 1989] Gennari, J. H., Langley, P., and Fisher, D. H. (1989). Models of incremental concept formation. In *Artificial Intelligence*, pages 11–61.

[Gilleron et al., 2006] Gilleron, R., Marty, P., Tommasi, M., and Torre, F. (2006). Extraction de relations dans les documents web. In *6èmes journées francophones d'Extraction et Gestion des Connaissances*, volume 1, pages 415–420.

[Guha et al., 1998] Guha, S., Rastogi, R., and Shim, K. (1998). CURE : an efficient clustering algorithm for large databases. In *ACM SIGMOD International Conference on Management of Data*, pages 73–84.

[Guha et al., 2000] Guha, S., Rastogi, R., and Shim, K. (2000). ROCK : A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5) :345–366.

[Hamerly and Elkan, 2003] Hamerly, G. and Elkan, C. (2003). Learning the k in k-means. *17th Annual Conference on Neural Information Processing Systems*, pages 281–288.

[Hastie and Tibshirani, 1996] Hastie, T. and Tibshirani, R. (1996). Discriminant anlysis by gaussian mixtures. *Journal of the Royal Statistical Society : Series B*, 58(1) :155–176.

[Hinneburg and Keim, 1998] Hinneburg, A. and Keim, D. A. (1998). An efficient approach to clustering in large multimedia databases with noise. In *Knowledge Discovery and Data Mining*, pages 58–65.

[Hinneburg and Keim, 1999] Hinneburg, A. and Keim, D. A. (1999). Cluster discovery methods for large data bases - from the past to the future. In *ACM SIGMOD International Conference on Management of Data*. Tutorial Session.

[Jain et al., 1999] Jain, A., Murty, M., and Flynn, P. (1999). Data clustering : a review. *ACM Computing Surveys*, 31(3) :264–322.

[Jain et al., 2000] Jain, A. K., Duin, R. P. W., and Mao, J. (2000). Statistical pattern recognition : A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1) :4–37.

[Jolliffe, 1986] Jolliffe, I. T. (1986). *Principal Component Analysis*. Springer Verlag, New-York.

[Jollois and Nadif, 2004] Jollois, F. and Nadif, M. (2004). Efficient version of EM for gaussian mixture model. *AISTA/IEEE International Conference on Advances in Intelligent Systems - Theory and Applications*.

[Kailing et al., 2004] Kailing, K., Kriegel, H.-P., and Kröger, P. (2004). Density-connected subspace clustering for high-dimensional data. In *SIAM International Conference on Data Mining*, pages 246–257.

[Karypis et al., 1999] Karypis, G., Han, E.-H. S., and NEWS, V. K. (1999). Chameleon : Hierarchical clustering using dynamic modeling. *Computer*, 32(8) :68–75.

[Law et al., 2004] Law, M., Figueiredo, M., and A.K.Jain (2004). Simultaneous feature selection and clustering using a mixture model. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 26(9) :1154–1166.

[Lian et al., 2004] Lian, W., Cheung, D. W., Mamoulis, N., and Yiu, S.-M. (2004). An efficient and scalable algorithm for clustering XML documents by structure. *IEEE transactions on Knowledge and Data Engineering*, 16(1) :82–96.

[Liu et al., 2000] Liu, B., Xia, Y., and Yu, P. S. (2000). Clustering through decision tree construction. In *9th International Conference on Information and Knowledge Managment*, pages 20–29.

[Meila, 2003] Meila, M. (2003). Comparing clusterings by the variation of information. *COLT 2003*, pages 173–187.

[Meir and Rätsch, 2003] Meir, R. and Rätsch, G. (2003). An introduction to boosting and leveraging. In Mendelson, S. and Smola, A., editors, *Advanced Lectures on Machine Learning*, number 2600 in LNAI, pages 119–184. Springer Verlag.

[Nagesh et al., 1999] Nagesh, H., Goil, S., and Choudhary, A. (1999). MAFIA : Efficient and scalable subspace clustering for very large data sets. Technical report, Northwestern University.

[Ng and Han, 1994] Ng, R. T. and Han, J. (1994). Efficient and effective clustering methods for spatial data mining. In Bocca, J., Jarke, M., and Zaniolo, C., editors, *20th International Conference on Very Large Data Bases*, pages 144–155.

[Nierman and Jagadish, 2002] Nierman, A. and Jagadish, H. V. (2002). Evaluating structural similarity in XML documents. In *5th International Workshop on the Web and Databases*.

[Nigam et al., 2000] Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3) :103–134.

[Parsons et al., 2004] Parsons, L., Haque, E., and Liu, H. (2004). Evaluating subspace clustering algorithms. In *Workshop on Clustering High Dimensional Data and its Applications, SIAM International Conference on Data Mining*, pages 48–56.

[Patrikainen and Meila, 2004] Patrikainen, A. and Meila, M. (2004). Comparing subspace clusterings. Technical Report UW-CSE-2004-10-01, Helsinki University of Washington.

[Pelleg and Moore, 2001] Pelleg, D. and Moore, A. (2001). Mixtures of rectangles : Interpretable soft clustering. In Brodley, C. and Danyluk, A., editors, *18th International Conference on Machine Learning*, pages 401–408.

[Pettie and Ramachandran, 2000] Pettie, S. and Ramachandran, V. (2000). An optimal minimum spanning tree algorithm. In *Automata, Languages and Programming*, pages 49–60.

[Procopiuc et al., 2002] Procopiuc, C. M., Jones, M., Agarwala, P. K., and Murali, T. M. (2002). A monte carlo algorithm for fast projective clustering. In *ACM SIGMOD International Conference on Management of Data*, pages 418–427.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5 : Programs for Machine Learning*. Morgan Kaufmann.

[Quinlan, 2004] Quinlan, J. R. (2004). Data mining tools see5 and c5.0.

[Sarafis et al., 2003] Sarafis, I. A., Trinder, P. W., and Zalzala, A. M. S. (2003). Towards effective subspace clustering with an evolutionary algorithm. In *IEEE Congress on Evolutionary Computation*.

[Schwartz, 1979] Schwartz, G. (1979). Estimating the dimension of a model. *The Annals of Statistics*, 6(2) :461–464.

[Steinbach et al., 2000] Steinbach, M., Karypis, G., and Kumar, V. (2000). A comparison of document clustering techniques. In *KDD Workshop on Text Mining*.

[Su and Chang, 2000] Su, M.-C. and Chang, H.-T. (2000). Fast self-organizing feature map algorithm. *IEEE-NN*, 11(3) :721–733.

[Talavera, 2000] Talavera, L. (2000). Dependency-based feature selection for clustering symbolic data. *Intelligent Data Analysis*, 4 :19–28.

[Termier et al., 2002] Termier, A., Rousset, M.-C., and Sebag, M. (2002). Treefinder : a first step towards XML data mining. In *IEEE International Conference on Data Mining*, pages 450–457.

[Torre, 1999] Torre, F. (1999). GloBo : un algorithme stochastique pour l'apprentissage supervisé et non-supervisé. In Sebag, M., editor, *Actes de la Première Conférence d'Apprentissage*, pages 161–168.

[Tsochantaridis et al., 2005] Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6 :1453–1484.

[Verma and Meila, 2003] Verma, D. and Meila, M. (2003). A comparison of spectral clustering algorithms. Technical report, University of Washington.

[Wang et al., 2002] Wang, H., Wang, W., Yang, J., and Yu, P. S. (2002). Clustering by pattern similarity in large data sets. In *ACM SIGMOD International Conference on Management of Data*, pages 394–405.

[Wang et al., 1997] Wang, W., Yang, J., and Muntz, R. R. (1997). STING : A statistical information grid approach to spatial data mining. In Jarke, M., Carey, M. J., Dittrich, K. R., Lochovsky, F. H., Loucopoulos, P., and Jeusfeld, M. A., editors, *23rd International Conference on Very Large Data Bases*, pages 186–195.

[Webb and Agar, 1992] Webb, G. I. and Agar, J. W. M. (1992). Inducing diagnostic rules for glomerular disease with the DLG machine learning algorithm. *Artificial Intelligence in Medicine*, 4 :419–430.

[Wolpert, 1992] Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5 :241–259.

[Woo and Lee, 2002] Woo, K. and Lee, J. (2002). *FINDIT : a fast and intelligent subspace clustering algorithm using dimension voting.* PhD thesis, Korea Advanced Institute of Science and Technology, Department of Electrical Engineering and Computer Science.

[Xu et al., 1998] Xu, X., Ester, M., Kriegel, H.-P., and Sander, J. (1998). A distribution-based clustering algorithm for mining in large spatial databases. In *14th International Conference on Data Engineering*, pages 324–331.

[Ye and Spetsakis, 2003] Ye, L. and Spetsakis, M. (2003). Clustering on unobserved data using mixture of gaussians. Technical report, York University, Toronto, Canada.

[Yip et al., 2003] Yip, K. Y., Cheung, D. W., and Ng, M. K. (2003). A highly-usable projected clustering algorithm for gene expression profiles. In *3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics*, pages 41–48.

[Zaki and Aggarwal, 2003] Zaki, M. J. and Aggarwal, C. C. (2003). XRules : An effective structural classifier for XML data. In *SIGKDD 03.*

[Zhang et al., 1997] Zhang, T., Ramakrishnan, R., and Livny, M. (1997). Birch : A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2) :141–182.

[Zhao and Karypis, 2002] Zhao, Y. and Karypis, G. (2002). Criterion functions for document clustering : Experiments and analysis. Technical Report TR #01–40, Department of Computer Science, University of Minnesota, Minneapolis, MN.

Liste des Algorithmes

Liste des tableaux

Table des figures

Summary

This phd thesis lies in the framework of unsupervised learning, that consists in creating different groups from a given dataset, so that data objects that are considered as the most similar are associated to the same group whereas data objects that are considered as different are associated to distinct groups. It thus allows us to extract some knowledge from this set of data.

We first propose two new methods that take into account the context in which the clusters are created, that is the fact that the caracteristics of the different groups may be defined according to different subsets of the attributes that describe the data. In the design of these methods, we also considered the problems of minimizing the prior knowledge required from the user and of presenting the results in an interpretable and visual way. We then present some possible extensions of these methods, in the framework of supervised learning and then faced with semi-structured data represented as trees.

Many experiments conducted on artificial as well as real data are presented and show the interest of these methods. Finally, as evaluating the results produced by an unsupervised learning approach and comparing such methods are open problems, we propose a new evaluation method that is more global, objective and quantitative than the traditionally used ones, and we experimentally show its relevance.

Keywords

- unsupervised learning
- subspace clustering
- knowledge extraction
- explanatory rules
- contextualization
- visualization
- cascade evaluation
- semi-structured data

Résumé

Cette thèse se place dans le cadre de l'apprentissage non supervisé, qui consiste à former différents groupes à partir d'un ensemble de données, de telle manière que les données considérées comme les plus similaires soient associées au même groupe et qu'au contraire les données considérées comme différentes se retrouvent dans des groupes distincts, permettant ainsi d'extraire de la connaissance à partir de ces données.

Nous proposons d'abord deux nouvelles méthodes qui prennent en compte le contexte dans lequel les groupes sont créés, c'est-à-dire le fait que les caractéristiques des différents groupes peuvent être définies sur différents sous-ensembles des attributs décrivant les données. Dans la mise en oeuvre de ces méthodes, nous avons également considéré les problématiques de la minimisation du nombre de connaissances a priori requises de la part de l'utilisateur et de la présentation des résultats sous forme compréhensible et visuelle. Nous présentons ensuite plusieurs extensions possibles de ces méthodes, dans le cadre de l'apprentissage supervisé puis face à des données semi-structurées représentées sous forme arborescente.

Différentes expérimentations sur données artificielles puis sur données réelles sont présentées qui mettent en avant l'intérêt de ces méthodes. Le problème de l'évaluation des résultats produits par une méthode d'apprentissage non supervisé, et de la comparaison de telles méthodes, restant aujourd'hui un problème ouvert, nous proposons enfin une nouvelle méthode d'évaluation plus globale, objective et quantitative que celles utilisées traditionnellement, et dont la pertinence est montrée expérimentalement.

Mots clés

– apprentissage non supervisé
– subspace clustering
– extraction de connaissances
– règles explicatives
– contextualisation
– visualisation
– évaluation en cascade
– données semi-structurées

Résumé

Cette thèse se place dans le cadre de l'apprentissage non supervisé, qui consiste à former différents groupes à partir d'un ensemble de données, de telle manière que les données considérées comme les plus similaires soient associées au même groupe et qu'au contraire les données considérées comme différentes se retrouvent dans des groupes distincts, permettant ainsi d'extraire de la connaissance à partir de ces données.

Nous proposons d'abord deux nouvelles méthodes qui prennent en compte le contexte dans lequel les groupes sont créés, c'est-à-dire le fait que les caractéristiques des différents groupes peuvent être définies sur différents sous-ensembles des attributs décrivant les données. Dans la mise en oeuvre de ces méthodes, nous avons également considéré les problématiques de la minimisation du nombre de connaissances a priori requises de la part de l'utilisateur et de la présentation des résultats sous forme compréhensible et visuelle. Nous présentons ensuite plusieurs extensions possibles de ces méthodes, dans le cadre de l'apprentissage supervisé puis face à des données semi-structurées représentées sous forme arborescente.

Différentes expérimentations sur données artificielles puis sur données réelles sont présentées qui mettent en avant l'intérêt de ces méthodes. Le problème de l'évaluation des résultats produits par une méthode d'apprentissage non supervisé, et de la comparaison de telles méthodes, restant aujourd'hui un problème ouvert, nous proposons enfin une nouvelle méthode d'évaluation plus globale, objective et quantitative que celles utilisées traditionnellement, et dont la pertinence est montrée expérimentalement.

Mots clés

– apprentissage non supervisé
– subspace clustering
– extraction de connaissances
– règles explicatives
– contextualisation
– visualisation
– évaluation en cascade
– données semi-structurées